arXiv:2602.15164v1 [cs.PL] 16 Feb 2026

从示例合成轨迹查询

Stephen Mell[1(✉)], Favyen Bastani[2], Steve Zdancewic[1], 和 Osbert Bastani[1]

[1] 宾夕法尼亚大学，美国宾夕法尼亚州费城 19104{sm1,stevez,obastani}@cis.upenn.edu[2] Allen人工智能研究所，美国华盛顿州西雅图 98104 favyenb@allenai.org

摘要。数据科学家通常需要编写程序来处理机器学习模型的预测结果，例如视频数据中的目标检测结果和轨迹。然而，由于现实世界数据具有模糊性，编写此类查询可能颇具挑战；尤其是，这些查询通常包含必须手动调优的实值参数。我们提出了一种名为 QUIVR 的新颖框架，可合成与给定示例集相匹配的轨迹查询。为了高效地合成参数，我们引入了一种剪枝参数空间的新颖技术，以及一种能进一步提升效率的新颖定量语义。我们在包含 17 项任务的基准测试上评估了 QUIVR，其中包括若干先前工作中的任务；结果表明，该框架能够为每项任务合成准确的查询，并且我们的优化显著缩短了合成时间。

1 引言

过去十年来，深度神经网络（DNN）成功解决了许多具有挑战性的人工智能问题 [47,70]。抽象地说，这些模型可以被视为真实世界数据的接口，例如，它们能够提供目标类别 [47,30], 检测结果 [60,59], 和轨迹 [11,83,10]。随后，这些预测会由程序进行处理，例如，用于识别驾驶模式 [5]、电视广播中的事件 [28],或动物行为 [67]。

然而，编写此类程序可能颇具挑战，因为它们仍然必须考虑真实数据的模糊性。为此，这些程序通常包含需要用户手动调优的实值参数。例如，考虑一个用于识别某辆汽车在另一辆汽车前方转弯的实例的汽车轨迹查询。该查询必须捕捉转弯汽车和穿过交叉路口的汽车各自的轨迹形状。此外，用户还必须选择从第一辆汽车变道到第二辆汽车穿过交叉路口之间的合适最大持续时间。即使是专家，也需要进行大量试验才能确定合适的参数值；根据我们的经验，为单个查询调优参数可能需要长达一小时。

我们关注的是用于查询对象跟踪器 [40,28,41,42,7,8,54,5]输出的轨迹数据库的程序。给定一个视频，跟踪器会预测每个视频帧中对象（例如汽车、人或小鼠）的位置，并关联连续视频帧中同一对象的检测结果。应用通常需要对这些轨迹进行后续分析。例如，在

自动驾驶中，遇到危险场景时，工程师通常会搜索更多符合该驾驶模式的示例，以改进规划器 [63,64,66]——例如，汽车行驶时距离过近 [82] ，或停在道路中间 [6]。目标跟踪也被用于跟踪机器人 [58,81], 和动物，以进行行为分析 [75,12,67], ，以及跟踪篮球运动员，以进行体育分析 [85,67]。

我们提出了一种算法，仅凭少量输入输出示例即可合成针对对象轨迹的查询。查询以轨迹的表示形式作为输入，即视频连续视频帧中的状态序列（例如位置、速度和加速度），并输出该轨迹是否符合其语义。我们的查询语言基于正则表达式——具体而言，查询是用户可扩展的一组谓词通过顺序组合、合取和迭代算子组合而成。例如，轨迹可能对应于视频中的汽车；图 1 展示了一个用于识别在交叉路口转弯的汽车的查询。正如我们在第 6 节中讨论的那样，完整的查询语言语义足够丰富，可以涵盖带测试的克林代数（KAT） [46]和信号时序逻辑（STL） [50]的（变体）；然而，实践中很少需要如此一般的表达能力，因此我们采用了一种精简的查询语言，使其在实际应用中表现良好。

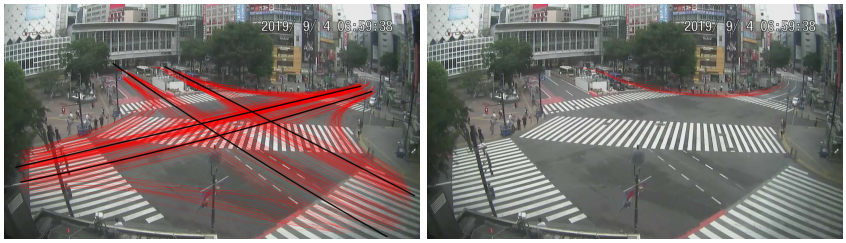
我们的算法通过对可能的查询空间进行枚举搜索，找出与给定示例一致的查询。在我们的设定中，一个关键挑战是谓词包含实值参数，这些参数也必须合成。因此，我们的策略是枚举草图，即只包含对应实值参数的空位的部分程序。对于每个草图，我们都会搜索实值参数空间，同时采用高效的剪枝策略来缩小搜索空间。总体而言，我们利用定量语义直接计算“边界参数”，即给定示例的标签从正类切换为负类时对应的参数。然后，根据目标标签，我们可以剪除边界参数一侧的整个搜索空间区域。我们证明，这种合成策略具有可靠性和（部分）完备性保证。

我们将接近方法实现为一个名为 Quivr 的系统。[1] 我们的实现侧重于固定位置摄像机拍摄的视频。虽然我们的语言和合成算法具有通用性，但我们设计的谓词针对特定场景。我们评估了 Quivr 在交通视频中识别驾驶模式的表现，其中包括受近期自动驾驶研究启发的模式； [64,63,66]， 在鼠标轨迹数据集中进行行为检测的表现； [72], 以及在时序逻辑合成文献中的一项合成任务上的表现。[44]我们展示了参数剪枝策略和查询评估优化如何显著缩短综合器的运行时间。

总而言之，我们的贡献如下：

– 一种用于查询对象轨迹的语言（第 3 节），以及一种根据示例合成此类查询的算法（第 4 节）。– 一种基于新颖定量语义的高效参数剪枝方法（第 4 节），相较于时序逻辑综合文献中最先进的定量剪枝技术，实现了 5.0× 倍加速。

[1] Quivr 代表视频轨迹查询归纳法。



(a) (b)

⟨在车道内1⟩ ; ⟨任意⟩; ⟨在车道内2⟩

图 1. (a) 交通摄像机拍摄的视频帧，以及对象轨迹（红色）和人工标注的车道（黑色）。(b) 查询选中的轨迹（下方）；该查询会选出在交叉路口转弯的汽车。

– 在 QUIVR 中实现我们的方法，并评估 QUIVR 在交通摄像机视频中识别驾驶行为，以及在小鼠轨迹数据集中识别小鼠行为的表现（第 5 节）；结果表明，其准确率显著优于神经网络基线。

2 概述

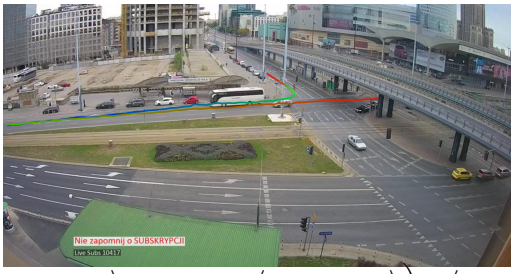
我们考虑这样一个假设场景：一位工程师正在为自动驾驶汽车设计控制算法，并希望在视频数据中识别某些驾驶模式。我们将展示如何使用我们的框架合成查询，以识别表现出特定行为的汽车轨迹。

视频数据。 交通摄像机是驾驶行为的丰富来源 [61,13,5]；我们在评估中使用的一个数据集是 YTStreams [7], ，其中包含来自多台此类摄像机的视频。图 1 (a) 展示了此类视频中的单个视频帧；我们使用了一个对象跟踪器 [83] 来识别所有汽车轨迹（红色）。

谓词。 QUIVR 假设已给定一组谓词，这些谓词与表现出感兴趣行为的部分轨迹相匹配；在合成过程中，它会考虑由这些谓词组成的查询。在图 1 (a) 中，工程师手动标注了此视频中感兴趣的车道（黑色），以指定四个 InLaneK 谓词，用于选择在视频中 K 可见、分别行驶在各条车道上的汽车轨迹。谓词可以通过实值参数进行配置。例如，

$$\langle \mathsf{InLane1} \rangle \wedge \langle \mathsf{DispLt}_\theta \rangle$$

搜索汽车在一段时间内保持在车道1上，且该时段开始和结束时的位移不超过 θ 的轨迹。请注意，原子谓词（如 $\langle \mathsf{DispLt}_\theta \rangle$）可以匹配多个时间步，而在正则表达式和时序逻辑等形式化方法中，原子谓词只作用于单个时间步。我们框架的一项关键特性是，



$$\Big(\big\langle \mathsf{InLane1}(A) \big\rangle\,;\ \langle 任意 \rangle;\ \big\langle \mathsf{InLane2}(A) \big\rangle\Big) \wedge \big\langle \mathsf{InLane2}(B) \big\rangle$$

图 2. 多对象查询（底部）的一次匹配结果（顶部）。该查询捕捉到一辆汽车 A 转入一条车道，跟在另一辆位于该车道上的汽车 B 后面。轨迹的颜色随时间从红色变为绿色。如图所示，右转的汽车是在直行汽车通过交叉路口后才转弯的。

可用谓词集合具有很强的可扩展性，用户也可以提供自己的谓词。有关我们在评估中使用的谓词，请参见第5.1节。

合成。 为了指定一种驾驶模式，工程师会提供少量轨迹的初始正例和负例；随后， QUIVR 合成一个能够正确标记这些示例的查询。在图 1 (b) 中，我们展示了执行所示查询的结果，该查询经合成后用于识别数据中的左转。通常，有多个查询都与初始示例一致。用户或许难以从视频中筛选正例，但通常很容易为给定轨迹添加标签。因此，为了消除歧义， QUIVR 会要求用户为其他轨迹添加标签 [62,19,36]。

多目标查询。 到目前为止，我们关注的是通过单独处理每条轨迹来识别轨迹的查询。我们的框架的一项关键特性是，用户可以针对多条轨迹表达查询，例如，

$$\big(\langle \mathsf{InLane1}(B) \rangle \wedge \langle \mathsf{ChangeLane2To1}(A) \rangle\big)\,;\, \langle \mathsf{InFront}(A, B) \rangle.$$

此查询表示汽车 B 位于车道1，而汽车 A 从车道2变道至车道1，汽车 A 最终位于汽车 B前方。请注意，谓词现在包含指明其所指对象的变量，并且谓词 $\mathsf{InFront}(A, B)$ 涉及多个对象。图 2 展示了多目标查询选出的一对轨迹示例。

3 查询语言

我们介绍一种用于匹配视频中对象轨迹的查询语言。我们的系统首先使用目标跟踪器对视频进行预处理，以获取轨迹，即由状态构成的序列 $z = (x_0,\ x_1,\ \ldots\ldots,\ x_{n-1})$ $x_i \in \mathcal{X}$。然后，我们语言中的查询 Q 会将每条轨迹 z 映射为一个值 $\mathbb{B} = \{0, 1\}$，用于指示它是否

匹配 z。我们的语言与信号时序逻辑和 KAT 都类似。一个关键区别是，谓词作用于 z 的任意子序列，而非单个状态 x。在正文中，我们考虑一种更简单的语言，但在附录 A 中，我们展示了如何扩展它，使其能够涵盖信号时序逻辑和 KAT。

轨迹。我们首先描述我们语言中查询的输入，即视频中一条或多条并发对象轨迹的表示。

考虑一个空间 $\mathcal{S}$ ，它对应于单个视频帧中的一次目标检测——例如，$s \in \mathcal{S} \subseteq \mathbb{R}^6$ 可以编码 s 在图像坐标中的二维位置、速度和加速度。当考虑 m 个并发对象时，令状态 空间为 $\mathcal{X} = \mathcal{S}^m$，则一条轨迹 $z \in \mathcal{Z} = \mathcal{X}^*$ 是长度为 $|z| = n$的状态序列$z = (x_0, x_1, ..., x_{n-1})$。我们使用记号 $z_{i:j} = (z_i, z_{i+1}, ..., z_{j-1})$来表示 z的子轨迹。

谓词。 我们假设给定一组谓词 Φ ，其中每个谓词 $\varphi \in \Phi$匹配轨迹 $z \in \mathcal{Z}$；我们使用$\mathsf{sat}_\varphi(z) \in \mathbb{B} = \{0, 1\}$ 来表示 φ匹配 z。如下文所述，我们语言中的查询将这些谓词组合起来，以匹配更复杂的模式。

接下来，我们语言中的谓词可以带有必须指定的实值参数。我们将带有参数 $\theta \in \mathbb{R}$ 的谓词 φ 记作 φ_θ。为了让我们的合成算法能够高效地合成这些实值参数，我们利用了查询中所用此类谓词的单调性。具体而言，我们假设这些谓词的语义具有如下形式

$$[\![\varphi_\theta]\!](z) \coloneqq \mathbb{1}(\iota_\varphi(z) \geq \theta),$$

其中 $\iota_\varphi : \mathcal{Z} \to \mathbb{R}$ 是评分函数。我们还假设 ι_φ 的取值范围有界（必要时可通过 Sigmoid 函数实现）。例如，对于谓词 DispLt_θ，我们有 $\iota_{\mathsf{DispLt}}(z) = -\|z_0 - z_{n-1}\|$。因此，$\iota_{\mathsf{DispLt}}(z) \geq \theta$ 表示总位移至多为 $-\theta$。我们在第 5.1 节介绍纳入的谓词；它们也可以轻松扩展。

语法。 我们语言的语法为

$$Q ::= \varphi \mid Q\,;\, Q \mid Q^k \mid Q \wedge Q,$$

其中 $Q^k = Q;Q;...;Q$ （k 次）。也就是说，基本情形是单个谓词 φ，查询可以通过顺序组合（Q；Q）和合取（$Q \wedge Q$）进行组合。析取、否定、克莱尼星号以及 STL 的“直到”运算符将在附录 A.2 中讨论。我们会在第 4.7 节介绍合成过程中对语言施加的约束。

语义。 查询的满足语义类型为 $[\![\cdot]\!] : \mathcal{Q} \to \mathcal{Z} \to \mathbb{B}$，其中 $\mathcal{Q}$ 是我们语言中所有查询的集合， $\mathcal{Z}$ 是轨迹的集合，且$\mathbb{B} = \{0, 1\}$。特别是， $[\![Q]\!](z) \in \mathbb{B}$ 表示查询 Q 是否与轨迹 z匹配。语义定义见图 3。单个谓词 φ 的基本情形用于检查 φ 是否与 z匹配；合取 $Q_1 \wedge Q_2$ 用于检查两个合取项是否都匹配；顺序组合 Q_1; Q_2 用于检查能否将 z 拆分为 $z = z_{0:k} z_{k:n}$，使得 Q_1 与 $z_{0:k}$ 匹配，且 Q_2 与 $z_{k:n}$匹配。该语义的求值时间为 $O(|Q| \cdot n^2)$。

$$\begin{aligned}
[\![\varphi]\!](z) &\coloneqq \mathsf{sat}_\varphi(z) \\
[\![Q_1 \wedge Q_2]\!](z) &\coloneqq [\![Q_1]\!](z) \wedge [\![Q_2]\!](z) \\
[\![Q_1 \,;\, Q_2]\!](z) &\coloneqq \bigvee_{k=0}^{n} [\![Q_1]\!](z_{0:k}) \wedge [\![Q_2]\!](z_{k:n})
\end{aligned}$$

图 3. 我们查询语言的满足语义； $z \in \mathcal{Z}$ 是长度为 n 的轨迹， $\varphi \in \Phi$ 是谓词。迭代（Q^k）可以表示为重复的顺序组合。

4 合成算法

我们介绍用于合成与给定示例集一致的查询的算法。该算法在可能的查询 [3]空间中执行语法引导枚举搜索。具体而言，它枚举草图，即仅缺少参数值的部分程序。对于每个草图，它使用定量剪枝策略，计算输入参数值的子集，使得由此得到的查询与给定示例一致。我们算法的一项关键贡献在于如何利用定量语义进行定量剪枝。

4.1 问题表述

部分查询。一个部分查询属于文法

$$Q ::= ?? \mid \varphi_{??} \mid \varphi \mid Q \,;\, Q \mid Q^k \mid Q \wedge Q.$$

注意，空位有两种：（i）谓词洞 $h =$??，可由子查询 Q填充；（ii）参数洞 $h = \varphi_{??}$ ，可由实数值 $\theta_h \in \mathbb{R}$填充。我们用 $\mathcal{H}_\varphi(Q)$ 表示 Q 的谓词洞，用 $\mathcal{H}_\theta(Q)$ 表示参数洞，并令 $\mathcal{H}(Q) = \mathcal{H}_\varphi(Q) \cup \mathcal{H}_\theta(Q)$。如果部分查询 Q 是草图[¯]（记作 $Q \in \mathcal{Q}_{\text{sketch}}$） [71] ，则 $\mathcal{H}_\varphi(Q) = \varnothing$；如果它是完整的（记作 $Q \in \mathcal{Q}$），则 $\mathcal{H}(Q) = \varnothing$。例如，对于 $Q = \langle \mathsf{DispLt}_{??1} \rangle \wedge ??2$，有 $\mathcal{H}_\theta(Q) = \{??1\}$，以及 $\mathcal{H}_\varphi(Q) = \{??2\}$。（我们用标识符 $i \in \mathbb{N}$ 标记每个空位 $h =??i$ ，以便区分。）

精化和补全。给定查询 $Q \in \mathcal{Q}$、谓词洞 $h \in \mathcal{H}_\varphi(Q)$ 和产生式 $R = Q \to f(Q_1, \ldots, Q_k)$，我们可以用 R 填充 h（记作 $Q' = \mathsf{fill}(Q, h, R)$）：将 h 替换为 $f(??1, \ldots, ??k)$，其中每个 $??i$ 都是一个新的空位。类似地，给定参数洞 $h \in \mathcal{H}_\theta(Q)$ 和值 $\theta_h \in \mathbb{R}$，我们称 Q' 为 Q 的一个子节点（记作 $Q \to Q'$）。接下来，如果存在一个序列 $Q \to \ldots \to Q''$，我们就称 Q'' 是 Q 的一个精化（记作 $Q \xrightarrow{*} Q''$）；如果此外还有 $Q'' \in \bar{\mathcal{Q}}$，则称其为 Q 的一个补全。例如，我们有

$$??1 \to ??2 \,;\, ??3 \to \langle \mathsf{InLane1} \rangle \,;\, ??3 \to \ldots.$$

这里，⟨InLane1⟩；??3 是 ??2 的一个子节点（也是精化）；??3 是通过用 $Q \to \langle$ InLane1$\rangle$ 填充 ??2 得到的，也就是说，

$$\langle \mathsf{InLane1} \rangle\,;\, ??3 = \mathsf{fill}(??2\,;\, ??3, ??2, Q \to \langle \mathsf{InLane1} \rangle).$$

参数。我们用 $\theta \in \mathbb{R}^{|\mathcal{H}_\theta(Q)|}$ 表示为每个 $h \in \mathcal{H}_\theta(Q)$选择的参数，用 $\theta_h \in \Theta_h \subseteq \mathbb{R}$ 表示空位 h的参数，并用 Q_θ 表示通过将每个 $h \in \mathcal{H}_\theta(Q)$填入 θ_h而得到的查询。注意，如果 $Q \in \mathcal{Q}_{\text{sketch}}$，那么 $Q_\theta \in \mathcal{Q}$ 就是完整的。例如，考虑这个草图

$$Q = \langle \mathsf{DispLt}_{??1} \rangle \wedge \langle \mathsf{MinLength}_{??2} \rangle.$$

这个查询有两个空位，因此其参数为 $\theta \in \mathbb{R}^2$。如果 $\theta = (3.2, 5.0)$，那么 $\theta_{??1} = 3.2$ 用于填入空位 ??1，$\theta_{??2} = 5.0$ 用于填入空位 ??2。具体来说，

$$Q_\theta = \langle \mathsf{DispLt}_{3.2} \rangle \wedge \langle \mathsf{MinLength}_{5.0} \rangle.$$

查询合成问题。给定示例 $W \subseteq \mathcal{W} = \mathcal{Z} \times \mathbb{B}$，其中 $\mathbb{B} = \{0, 1\}$，我们的目标是找到一个能够正确标记这些示例的查询 $Q \in \mathcal{Q}$ ——即，

$$\psi_W(Q) \coloneqq \bigwedge_{(z,y) \in W} (\llbracket Q \rrbracket(z) = y).$$

因此，$\psi_W(Q)$ 表示 Q 是否与已标注示例 W 一致。我们的目标是设计一种可靠且完备的合成算法——也就是说，当且仅当存在满足 $\psi_W(Q) = 1$ 的查询时，该算法才会找到这样的查询。

4.2 算法概述

我们的算法会枚举草图 $Q \in \mathcal{Q}_{\text{sketch}}$；对于每个草图，它都会尝试计算参数值 θ，使得补全后的查询 Q_θ 与 W 一致——也就是说，$\psi_W(Q_\theta) = 1$。找到一致查询后，算法可以立即停止，也可以继续找出与 W 一致的其他查询。算法 1 展示了这一高层策略——在每次迭代中，它会选择一个草图 Q，确定参数空间中的一个区域 B，该区域包含一致参数 $\theta \in B$，并将 (Q, B) 添加到一致查询列表中，以解决合成问题。

关键挑战在于：对于给定的草图 Q，如何在连续参数 θ 的空间中搜索，使得 Q_θ 与 W 一致。为了提高效率，我们大量依赖对搜索空间进行剪枝。概括来说，考虑在单个示例 $(z, y) \in W$ 上评估单个候选参数 θ——也就是检查 $\llbracket Q_\theta \rrbracket(z) = y$ 是否成立。如果该条件不成立，那么我们不仅可以将 θ 从搜索空间中剪枝，还可以剪去大量其他候选项。例如，假设 $\llbracket Q_\theta \rrbracket(z) = 1$，但 $y = 0$；如果 $\theta' \leq \theta$（在所有分量中），那么根据我们针对语义证明的单调性性质，还可以推出 $\llbracket Q_{\theta'} \rrbracket(z) = 1$。因此，我们也可以将 θ' 剪枝。

先前的工作利用这一特性来缩减搜索空间 [49,78,53]。对于给定示例 $(z, y) \in W$，我们可以采用基于二分搜索的策略，

算法 1 使用算法 2 中的子程序合成一致查询

过程 SynthesizeQuery(W)
  $\mathcal{Q}_{\text{con}} \leftarrow \varnothing$
  对 $Q \in \mathcal{Q}_{\text{sketch}}$ **执行**
    $B \leftarrow$ 合成参数(W, Q)
    $\mathcal{Q}_{\text{con}} \leftarrow \{(Q,\ B)\}$
  返回 $\mathcal{Q}_{\text{con}}$

以 ε 的精度，在 O（log($1/\varepsilon$)）个步骤内识别“边界”参数 θ ——即计算满足 $[\![Q_{\theta-\vec{\varepsilon}}]\!](z) = 1$ 且 $[\![Q_{\theta+\vec{\varepsilon}}]\!](z) = 0$ 的 θ。

我们的算法避免了这一二分搜索过程，在实践中可以显著提升速度。其核心思想是为查询设计一种定量语义，直接计算 θ；事实上，这种定量语义与鲁棒时序逻辑语义密切相关，在后者中，满足语义里的合取和析取分别由最小值和最大值取代。

4.3 使用边界参数进行剪枝

我们首先介绍如何利用“边界参数”来剪枝参数搜索空间的一部分。首先，对于任意候选参数θ，我们可以剪去参数 $\theta' \le \theta$ （如果 $[\![Q_\theta]\!](z) = 1$ 且 $y = 0$）或 $\theta' \ge \theta$ （如果$[\![Q_\theta]\!](z) = 0$ 且 $y = 1$）。参数空间中被剪去的区域呈超矩形形状，我们称之为盒子。为方便起见，令 $\vec{\infty} := (\infty, \ldots, \infty)$。

定义 1。给定 $x,\ y \in \bar{\mathbb{R}}^d$，其中 $\bar{\mathbb{R}} = \mathbb{R} \cup \{\pm\infty\}$，**盒子** 是一个轴对齐半开超矩形 $\lfloor x,\ y \rceil := \{v \mid x_i < v_i \le y_i\} \subseteq \mathbb{R}^d$。

确保参数能够剪枝搜索空间中盒子的关键性质是，语义关于 θ 单调递减。

引理 1。给定草图 Q、轨迹 z 以及两个候选参数 θ、$\theta' \in \mathbb{R}^d$，使得 $\theta \le \theta'$ 按分量比较成立，我们有 $[\![Q_\theta]\!](z) \ge [\![Q_{\theta'}]\!](z)$。

证明遵循对查询语义进行结构归纳法：基础情形成立，因为谓词的语义 $\mathbb{1}(\iota_\varphi(z) \ge \theta_k)$ 关于 θ_k 单调递减；归纳情形成立，因为合取和析取关于其输入单调递增（因此它们关于 θ_k 也单调递减）。下面我们将展示，若找到边界参数，单调性如何确保我们能够剪枝搜索空间中的整个区域。

例如，假设我们有两条轨迹：z_0 一辆汽车先快速行驶、然后缓慢行驶，z_1 一辆汽车先缓慢行驶、然后快速行驶；我们尝试为 $W = \{(z_0,\ 0),\ (z_1,\ 1)\}$合成一个查询。为简单起见，假设 $z_0 = (0。9,\ 0。6)$ 和 $z_1 = (0。5,\ 0。8)$ 都只有两个时间步，且每个时间步仅包含一个表示速度的分量。此外，假设只有一个谓词 $\langle \text{VelGt}_\theta \rangle$，它匹配速度至少为 θ 的时间步，

其中 θ 是一个实数参数。由于 $\langle \text{VelGt}_\theta \rangle$ 匹配单个时间步，因此其满足语义除了在长度为 1 的轨迹上之外均为 0，所以:

$$\iota_{\text{VelGt}}((z_0)_{0:1}) = 0.9 \qquad \iota_{\text{VelGt}}((z_0)_{1:2}) = 0.6 \qquad \iota_{\text{VelGt}}((z)_{i:i}) = -\infty$$
$$\iota_{\text{VelGt}}((z_1)_{0:1}) = 0.5 \qquad \iota_{\text{VelGt}}((z_1)_{1:2}) = 0.8 \qquad \iota_{\text{VelGt}}((z)_{0:2}) = -\infty$$

考虑草图 $Q = \langle \text{VelGt}_{??1} \rangle; \langle \text{VelGt}_{??2} \rangle$。我们可以看到，候选参数 (0.5,0.6) 满足 $[\![Q_{(0.5,0.6)}]\!](z_1) = 1$:

$$\begin{aligned}
[\![Q_{(0.5,0.6)}]\!]((z_1)_{0:n}) &= \bigvee_{k=0}^{2} [\![\langle \text{VelGt}_{0.5} \rangle]\!]((z_1)_{0:k}) \wedge [\![\langle \text{VelGt}_{0.6} \rangle]\!]((z_1)_{k:n}) \\
&= [\![\langle \text{VelGt}_{0.5} \rangle]\!]((z_1)_{0:1}) \wedge [\![\langle \text{VelGt}_{0.6} \rangle]\!]((z_1)_{1:2}) \\
&= \mathbb{1}(0.5 \geq 0.5) \wedge \mathbb{1}(0.8 \geq 0.6) \\
&= 1,
\end{aligned}$$

其中第二个等式成立，是因为 $\langle \text{VelGt}_\theta \rangle$ 仅匹配长度为 1 的轨迹，因此 $k = 0$ 和 $k = 2$ 的情形求值为 0。由于语义是单调递减的，对于任意 $\theta \in \lfloor (-\infty, -\infty), (0.5,0.6) \rceil$，我们有 $[\![Q_\theta]\!](z_1) = 1$。

然而请注意，如果我们把任意 $\vec{\varepsilon} > 0$ 向上移动，就会得到$[\![Q_{(0.5+\varepsilon_1, 0.6+\varepsilon_2)}]\!](z_1) = \mathbb{1}(0.5 \geq 0.5 + \varepsilon_1) \wedge \mathbb{1}(0.8 \geq 0.6 + \varepsilon_2) = 0$。因此我们知道，对于任意 $\theta \in \lfloor (0.5, 0.6), (\infty, \infty) \rceil$，有$[\![Q_\theta]\!](z_1) = 0$。这是因为 (0.5,0.6) 位于 $\{\theta' \mid [\![Q_{\theta'}]\!](z) = 0\}$ 与 $\{\theta' \mid [\![Q_{\theta'}]\!](z) = 1\}$ 之间的边界上。该边界在我们的算法中起着关键作用。

定义 2. 给定一个带有 d 个参数空位的草图 Q 和一条轨迹 z，如果满足以下任一条件，我们就称 $\theta \in \mathbb{R}^d \cup \{\bot, \top\}$ 为一个边界参数：

- $\theta \in \mathbb{R}^d$ 且 $[\![Q_\theta]\!](z) = 1$，但对于所有 $\theta' \in \lfloor \theta, \vec{\infty} \rceil$，$[\![Q_{\theta'}]\!](z) = 0$；
- $\theta = \bot$ 且对于所有 $\theta' \in \lfloor -\vec{\infty}, \vec{\infty} \rceil$，$[\![Q_{\theta'}]\!](z) = 0$；
- $\theta = \top$ 且对于所有 $\theta' \in \lfloor -\vec{\infty}, \vec{\infty} \rceil$，$[\![Q_{\theta'}]\!](z) = 1$

在第一种情况下，根据单调性，对于所有 $\theta' \in \lfloor -\vec{\infty}, \theta \rceil$，我们还可以得出 $[\![Q_{\theta'}]\!](z) = 1$；因此，$\theta$ 位于参数 θ' 的边界上，其中 $Q_{\theta'}$ 的取值为 1，而另一侧的取值为 0。第二种和第三种情况分别是 $Q_{\theta'}$ 始终取值为 0 和 1。

给定示例 $(z, y) \in W$的一个边界参数 θ，若 $y = 1$，我们可以剪枝$\lfloor \theta, \vec{\infty} \rceil$；若 $y = 0$，则可以剪枝 $\lfloor -\vec{\infty}, \theta \rceil$。直观而言，边界参数沿参数空间中的固定方向提供最优剪枝。因此，我们的算法侧重于计算用于剪枝的边界参数。

在图4 (a)中，如果 θ_1 是 z_1的边界参数，我们知道蓝色区域满足 z_1，因此与标签1一致；而红色区域不满足 z_1，因此与标签1不一致。同样，在图4 (b)中，如果 θ_0 是 z_0的边界参数，我们知道红色区域满足 z_1，因此与标签0不一致；而蓝色区域不满足 z_0，因此与标签0一致。

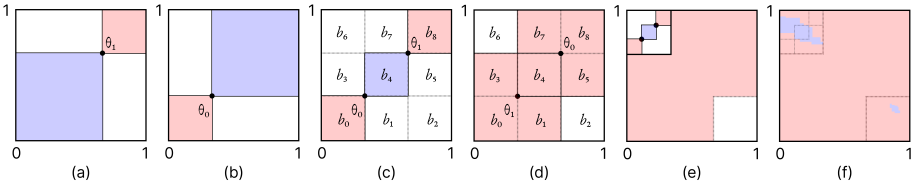


图 4. (a) 显示了 z_1 的一个边界参数 θ_1，以及与 z_1 不一致、可以剪枝的区域（红色），还有与之相符的区域（蓝色）。(b) 类似地显示了 z_0 的一个边界参数 θ_0。(c) 显示了由 θ_0 和 θ_1 组成的剪枝对、与两者都相符的区域（蓝色），以及与其中任一者不一致的区域（红色）。(d) 与 (c) 相同，但如果 θ_0 和 θ_1 交换位置。标签 b_0 至 b_8 表示 (c) 和 (d) 中对应的盒子。(e) 显示了如果 (d) 是搜索的第一步结果，且接下来选择 b_6，搜索可能如何继续。(f) 显示了搜索过程在 (d) 和 (e) 中可能趋近的、与真值相符（蓝色）和不一致（红色）的区域。

4.4 使用边界参数对进行剪枝

为了将剪枝扩展到整个数据集 W，我们可以直接对每个$(z,\ y)\ \in W$对应的剪枝区域取并集并进行剪枝。不过，我们的方法的一个重要特点是，还可以确定参数空间中的一些区域，在这些区域内可以保证参数与 W一致。为了将这一想法形式化，我们引入“剪枝对”的概念，它是一对边界参数，可能有助于我们找到这样的一致区域。

定义 3. 给定草图 Q 和数据集 W，一对边界参数$\theta^-,\ \theta^+ \in \mathbb{R}^d \cup \{\bot, \top\}$ 是 Q 和 W 的剪枝对，当且仅当满足以下所有条件：

– θ^+ 是某个 $z \in W^+$ 的边界参数，并且对于所有满足 $z' \neq z$的 $z' \in W^+$，都有 $[\![Q_{\theta^+}]\!](z') = 1$。– θ^- 是某个 $z \in W^-$ 的边界参数，并且对于所有满足 $z' \neq z$的 $z' \in W^-$，都有 $[\![Q_{\theta^-}]\!](z') = 0$。– $\theta^- < \theta^+$ 或 $\theta^- \geq \theta^+$。

如果 $\theta^- < \theta^+$，则剪枝对（$\theta^-,\ \theta^+$）是一致的；否则是不一致的。

我们的算法沿固定方向搜索剪枝对——也就是说，它考虑一条曲线 $L \subseteq \mathbb{R}^d$，并沿着 L 寻找下一个剪枝对：

$$\theta^+ = \sup\left\{\theta \in L \,\middle|\, \bigwedge_{z\in W^+} [\![Q_\theta]\!](z) = 1\right\},\quad \theta^- = \inf\left\{\theta \in L \,\middle|\, \bigwedge_{z\in W^-} [\![Q_\theta]\!](z) = 0\right\}.$$

直观来说，θ^+ 是能够正确分类所有正例的最大 θ，反之亦然，θ^-也是如此。我们将范围限制为各个分量都单调递增的曲线 L；在这种情况下，上确界和下确界都有定义，因为L 带有一种与 $\mathbb{R}^d$上的标准偏序一致的全序（从最小点到最大点）。那么，$(\theta^-,\ \theta^+)$构成一个剪枝

对：由于 θ^+ 是 z的边界参数，如果我们取任意更大的 θ^+ ，那么对于某个 $z \in W^+$，必有 $[\![Q_\theta]\!](z) = 0$ ， θ^-的情况也类似。

给定曲线 L，我们可以通过二分搜索近似计算 θ^+ 和 θ^- 。然而，我们的算法通过使用定量语义直接计算 θ^+ 和 θ^- ，避免了这一步，相关内容将在第 4.6 节中介绍。

图4 (c) 展示了边界参数对 z_0 的 θ_0 和对 z_1 的 θ_1 （其中 L 是对角线）如何剪枝参数空间。蓝色区域必定与 W 一致，因为它是 θ^+ 下方区域与 θ^- 上方区域的交集；前者必须满足 $[\![Q_\theta]\!](z_1) = 1$，后者必须满足 $[\![Q_\theta]\!](z_0) = 0$。反之，红色区域与 z_0 或 z_1 不一致，因此也与 W 不一致。因此，可以剪去红色区域，而蓝色区域则是我们合成问题的解。注意，红色区域是图4 (a) 和 (b) 中红色区域的并集，而蓝色区域是图4 (a) 和 (b) 中蓝色区域的交集。

这一模式适用于任意一致剪枝对（$\theta^- < \theta^+$）；如果该剪枝对不一致（$\theta^- \geq \theta^+$），则所得模式如图4 (d) 所示；在这种情况下，我们可以像之前一样剪枝红色区域，但不存在包含解的蓝色区域。一般而言，对于一个 d 维参数空间，剪枝对会将参数空间划分为 3^d 个盒子（也就是说，对于每个维度，盒子可以低于、与中心盒齐平或高于中心盒）。低于 θ^- 和高于 θ^+ 的区域可以剪枝，而 θ^- 与 θ^+ 之间的区域（如果存在）包含合成解。具体而言，根据定义和单调性可知：

引理 2. 每个 $\theta \in \lfloor -\infty, \theta^- \rceil$ 和 $\theta \in \lfloor \theta^+, \infty \rceil$ 都与 W不一致，而每个 $\theta \in \lfloor \theta^-, \theta^+ \rceil$ 都与 W一致。

剩余的盒子需要由我们的算法进一步分析。

4.5 剪枝参数搜索算法

接下来，我们介绍算法 2。该算法在参数空间中搜索，以便为给定的数据集 W 填充草图 Q 。该算法使用一个子程序，该子程序接收一个盒子，并返回该盒子中的一个剪枝对；我们将在第 4.6 节介绍该子程序。给定这个子程序后，我们的算法会维护一个包含“未知”盒子的工作列表（即，尚不确定这些盒子中的参数与 W 是一致还是不一致）。每次迭代时，它会从工作列表中取出一个盒子（按先进先出顺序），使用给定的子程序在该盒子中查找一个剪枝对，应用上一节所述的剪枝过程，然后将每个新出现的未知盒子加入工作列表。

最后一步中，当前盒子 b 被分成 3^d 个更小的盒子。如果点对 (θ^-, θ^+) 不一致，则剪枝盒子 $b_{\text{center}} := \lfloor \min\{\theta^-, \theta^+\}, \max\{\theta^-, \theta^+\} \rceil$ （加入 B_{inc}）；否则，它包含合成问题的解（加入 B_{con}）。盒子 $b_{\text{lower}} = \lfloor -\infty, \min\{\theta^-, \theta^+\} \rceil$ 和 $b_{\text{upper}} = \lfloor \max\{\theta^-, \theta^+\}, \infty \rceil$ 始终会被剪枝。盒子 $b \in B_{\text{incomp}}$ 是 b的其余角点，其一致性始终无法判定（加入 B_{unk}）。

算法 2 为给定草图合成一致参数

过程 SynthesizeParameters（W， Q）
  $B_{\text{con}} \leftarrow \varnothing$； $B_{\text{inc}} \leftarrow \varnothing$、 $B_{\text{unk}} \leftarrow \{b_{\text{initial}}\}$
  对于 $i \in \{1, ..., N\}$，**执行**
    $b \leftarrow$弹出(B_{unk})
    θ^-， $\theta^+ \leftarrow$ 计算剪枝对(W， Q， b)
    $b_{\text{center}} \leftarrow \lfloor$最小值$\{\theta^-, \theta^+\}$，最大值$\{\theta^-, \theta^+\} \rceil$
    b_{lower}， b_{upper}， B_{incomp}， $B_{\text{extra}} \leftarrow$ 拆分盒子(b， b_{center})
    如果 $\theta^- < \theta^+$ **则**
      $B_{\text{con}} \leftarrow B_{\text{con}} \cup \{b_{\text{center}}\}$
      $B_{\text{inc}} \leftarrow B_{\text{inc}} \cup \{b_{\text{lower}}, b_{\text{upper}}\}$
      $B_{\text{unk}} \leftarrow B_{\text{unk}} \cup B_{\text{incomp}} \cup B_{\text{extra}}$
    否则如果 $\theta^- \geq \theta^+$ **则**
      $B_{\text{inc}} \leftarrow B_{\text{inc}} \cup \{b_{\text{center}}, b_{\text{lower}}, b_{\text{upper}}\} \cup B_{\text{extra}}$
      $B_{\text{unk}} \leftarrow B_{\text{unk}} \cup B_{\text{incomp}}$
  返回 B_{con}

如果 (θ^-, θ^+) 一致，则剩余盒子 $b \in B_{\text{extra}}$ 的一致性无法判定；否则则不一致。在我们的示例中，如果算法的第一步得到图4 (d)，那么第二步可能会取出 b_6 ，并得到图4 (e)。

以下可靠性结果可直接由引理 2 得出。

定理1. 在算法 *2* 中，每个 $\theta \in B_{\text{con}}$ 都与 W 在 Q上保持一致，且每个 $\theta \in B_{\text{inc}}$ 都不一致。

此外，对于几乎所有参数，该算法都是完备的：

定理 2. $\{\theta \in b \mid b \in B_{\text{unk}}\} \to 0$ 的勒贝格测度为 $N \to \infty$。

证明见附录D.1。换言之，除去一个测度为零的子集，所有参数最终都会被判定为一致或不一致；直观而言，可能永远无法被判定的参数位于决策边界上。该结果成立，是因为在任意搜索深度下，参数空间中被剪枝部分的比例都有一个下界。

4.6 通过定量语义计算剪枝对

剪枝算法依赖于这样一种能力：给定一个盒子 b，能够在参数空间限制到 b 后计算出一个剪枝对 (θ^-, θ^+)。回想一下， θ^+ 必须是某个 $z^+ \in W^+$ 的边界参数，并且必须满足$[\![Q_{\theta^+}]\!](z) = 1$ ，对于所有其他的 $z \in W^+$均如此；而 θ^- 必须是某个 $z^- \in W^-$的边界参数，并且必须满足 $[\![Q_{\theta^-}]\!](z) = 0$，对于所有其他的 $z \in W^-$均如此。

给定一个盒子 $b = \lfloor \theta_{\min}, \theta_{\max} \rceil$，我们的算法将从 $\theta_{\min}$ 到 $\theta_{\max}$ 的对角线作为 $L \subseteq \mathbb{R}^d$ ，并沿着 L计算剪枝对。我们可以简单地使用二分搜索：对于 θ^+，我们搜索使 $\bigwedge_{z \in W^+} [\![Q_\theta]\!](z)$ 从 0 变为 1 的参数，对于 θ^- 和 $\bigwedge_{z \in W^-} \neg [\![Q_\theta]\!](z)$ 也同样处理。

相反，借助定量语义，我们可以直接计算θ^+ 和 θ^-，从而大幅缩短计算时间。给定一个草图

$$
\begin{aligned}
[\![\varphi_{??i}]\!]^q_{v,u}(z) &\coloneqq \frac{\iota_\varphi(z) - v_i}{u_i} \\
[\![\varphi]\!]^q_{v,u}(z) &\coloneqq \begin{cases} \infty & \text{if } \mathsf{sat}_\varphi(z) = 1 \\ -\infty & \text{if } \mathsf{sat}_\varphi(z) = 0. \end{cases} \\
[\![Q_1 \wedge Q_2]\!]^q_{v,u}(z) &\coloneqq \min\{[\![Q_1]\!]^q_{v,u}(z), [\![Q_2]\!]^q_{v,u}(z)\} \\
[\![Q_1\,;\,Q_2]\!]^q_{v,u}(z) &\coloneqq \max_{0\le k\le n} \min\{[\![Q_1]\!]^q_{v,u}(z_{0:k}), [\![Q_2]\!]^q_{v,u}(z_{k:n})\}
\end{aligned}
$$

图 5. 我们语言的定量语义，以草图 Q、轨迹 z、参数 $v \in \mathbb{R}^d$ 和正向量 $u \in \mathbb{R}^d_{>0}$ 为输入。n 是 z 的长度。

Q、轨迹 z、参数 $v \in \mathbb{R}^d$和正向量 $u \in \mathbb{R}^d_{>0}$，我们设计了一种¯定量语义$[\![Q]\!]^q_{v,\ u}(z) \in \mathbb{R}$ 使得参数$[\![Q]\!]_{v,\ u}(z) \cdot u + v$是一个边界参数。直观而言，该语义计算从 v出发，需要经过多少个大小为 u的步骤才能到达边界。（在我们的算法中，步骤数始终属于 $[0, 1]$。）然后，对于一个盒子 $b \coloneqq [\theta_{\min},\ \theta_{\max}]$，我们可以取 $v = \theta_{\min}$ 和 $u = \theta_{\max} - \theta_{\min}$，并计算

$$
\theta^+ = \left(\min_{z \in W^+} [\![Q]\!]^q_{v,u}(z)\right) \cdot u + v, \qquad \theta^- = \left(\max_{z \in W^-} [\![Q]\!]^q_{v,u}(z)\right) \cdot u + v.
$$

我们在图 5 中定义定量语义。$\varphi_{??}$ 的基本情况通过 v 和 u 对 ι_φ 进行调整和重新缩放，而其他情况则将满足语义中的合取与析取替换为最小值和最大值。我们有如下关键结果（其中 $\infty \cdot u \coloneqq \top$、$-\infty \cdot u \coloneqq \bot$、$\top + v \coloneqq \top$ 以及 $\bot + v \coloneqq \bot$）：

定理 3. 对于草图 Q、轨迹 z、参数 $v \in \mathbb{R}^d$ 和正向量 $u \in \mathbb{R}^d_{>0}$，我们有 $[\![Q]\!]^q_{v,\ u}(z) \cdot u + v$ 是 z 对于 Q 的边界参数。

完整证明见附录 D.2。为了直观起见，考虑 $\theta_{\min} = \vec{0}$ 和 $\theta_{\max} = \vec{1}$（即当前盒子 $b \subseteq \mathbb{R}^d$ 是单位超立方体）。此时，$v = \vec{0}$ 和 $u = \vec{1}$，因此 $[\![Q]\!]^q_{v,\ u}$ 可化简为时序逻辑 [25]的标准最大-最小定量语义。

现在，考虑基本谓词 $[\![\varphi_{\theta_i}]\!] = \mathbb{1}(\iota_\varphi(z) \ge \theta_i)$的满足语义，那么语义发生翻转时 θ_i 的值就是 $\iota_\varphi(z)$。因此，任何第 i个分量为 $\iota_\varphi(z)$ 的参数都是边界参数；又因为L 在所有维度上的斜率都相同，沿 L 方向的边界参数为$\iota_\varphi(z) \cdot \vec{1} + \vec{0} = [\![\varphi_{??i}]\!]^q_{\vec{0},\ \vec{1}}(z) \cdot \vec{1} + \vec{0}$。

在归纳情形中，只需证明我们可以在语义中用最小值和最大值替换合取和析取。由于满足语义是单调递减的，沿着 L 向上移动时，我们会在某个时刻从 1 转变为 0。只要任一合取项变为 0，合取就会变为 0，因此转变会发生在遇到最先发生转变的合取项时（即它们转变点中的最小值）。与此相应，只有当两个析取项都变为 0 时，析取才会变为 0，因此我们会在最后一个析取项发生转变时转变（即它们转变点中的最大值）。

析取项变为 0，因此我们会在最后一个析取项的转变点（即它们中的最大值）处发生转变。

最后，u 和 v 背后的直觉是，它们会对参数进行“预处理”，使我们沿着盒子 $\lfloor v,\ v+u \rceil$ 的对角线进行评估，而不是 $\left\lfloor \vec{0},\ \vec{1} \right\rceil$。

4.7 实现

我们在一个名为 QUIVR的系统中实现了我们的方法。该系统首先针对少量已标注示例运行算法1。

主动学习。 对于少量示例，通常存在许多与标签一致、但对其余数据的标签意见不一的查询。为消除歧义，我们采用主动学习策略，请用户标注我们选定的特定轨迹，然后将这些轨迹添加到已标注示例集中。与新标签不一致的查询会被丢弃。标注过程持续进行，直到所有一致查询对全部未标注数据的标签达成一致。

选择接下来要查询用户的轨迹 z^* 时，我们会选取一致查询集合 C 意见分歧最大的那条轨迹——即，

$$z^* = \arg\min_{z \in Z} \left| J(z) - \frac{1}{2} \right|,$$

其中

$$J(z) \coloneqq |C|^{-1} \sum_{Q_\theta \in C} \mathbb{1}\left(\psi_{(z,y)}(Q_\theta)\right)$$

是对轨迹 z预测为正类标签的一致查询所占的比例。

搜索实现。 在某些情况下，搜索一致参数可能会耗费很长时间。为提高性能，我们设置了超时：对于每个草图，如果满足以下任一条件，就暂停搜索：(i) 找到某个一致参数框，或 (ii) 已超过 25 个步骤。在这两种情况下，我们都会保存一致、不一致和未知参数框的集合。在主动学习的每一步中，新标注的示例可能会使先前一致的参数变得不一致，因此我们将所有一致参数框标记为未知。随后恢复搜索，直到再次满足以下任一条件：(i) 找到某个一致参数框（可能是之前找到的同一个），或 (ii) 再次超过 25 个步骤。

请注意，虽然这个超时可能会导致我们比严格必要的频率更高地向用户查询，但由于我们会在向用户查询后继续搜索，因此不会影响我们方法的可靠性或完备性。

完整查询选择。 主动学习和 F_1 分数评估（见第 5 节）都需要带有特定参数的完整查询，而不是带有参数盒子的草图。由于一致查询集合 C 是无限大的，因此我们改为对每个已知具有一致参数的草图使用一个查询（搜索超时的草图不包括在内）。对于这些草图，我们选取已知一致参数盒子的中点。

表1. YTStreams 数据集使用的谓词。

谓词	描述
InLaneK(A)	在区间的每个时间步，对象 A 是否足够接近车道K的标注曲线，且 A 的运动方向是否足够符合 K 的曲线。
持续时间不短	区间是否至少持续 5 秒。
平均加速度大于θ	区间上的平均加速度是否至少为 θ。
距离小于θ	在区间内的每个时间步，两个之间的距离是否对象小于 θ。
速度比大于$\theta(A, B)$	在区间内的每个时间步，A 的速度除以 B 的速度至少为 θ。
距离小于$\theta(A)$	区间第一视频帧中的位置之间的距离是否以及最后一个视频帧中的位置之间的距离小于 θ。

5 评估

我们展示了如何利用我们的方法合成查询来解决有趣的任务：具体而言，我们表明，(i) 仅凭少量初始示例，它就能合成在留出测试集上表现良好的查询；(ii) 我们的优化显著缩短了合成时间。

5.1 实验设置

数据集。 我们在两个对象轨迹数据集上进行评估：YTStreams [7],由固定位置交通摄像机拍摄的视频及提取的对象轨迹组成；MABe22 [72], 由实验室环境中最多三只小鼠互动的轨迹组成。我们还在 STL合成文献 [44]中的一项合成海上监控任务上进行评估。在 YTStreams 上，我们使用两台交通摄像机，一台位于东京，另一台位于华沙，并考虑单辆汽车或汽车对。在 MABe22 上，我们考虑小鼠对。所用谓词请参见表1（YTStreams）、表5（MABe22）和表6（海上监视）。

任务。 在 YTStreams 上，我们手动编写了 5 个真实查询。多个查询适用于多种配置（例如，不同的车道对），因此总共有 10 个查询（表2中的任务 H-Q）。实值参数是通过目视检查所选轨迹是否符合预期而手动确定的。这些查询涵盖了广泛的行为；例如，它们能够捕捉人类驾驶员进行无保护转弯等行为——这对自动驾驶汽车而言是一项重要挑战 [64], ，也能捕捉汽车试图超车的行为 [66]。图 6 展示了三个多目标查询所选轨迹的示例。MABe22 描述了 9 个具有科学研究价值的小鼠行为查询。我们实现了其中最复杂的 6 个，将其用作真实查询（表7中的任务 A-F）。海上监视任务带有轨迹标签，因此不需要真实查询（任务 G）。

合成。 对于每个任务，我们将所有轨迹的集合 Z 划分为训练集 Z_{train} 和测试集 Z_{test}，使用视频前半部分的轨迹

表2。 YTStreams 数据集的真值查询。“ID”表示哪些任务是某个查询的实例。多个实例化对应于“车道1”和“车道2”使用不同车道的情况。第一个是一个对象的涩谷查询，第二个是一个对象的华沙查询，其余是两个对象的华沙查询。

IDs	查询
H、I、J、K	匹配从车道1出发并在车道2结束的转弯汽车。 $\langle \text{InLane1}(A) \rangle$；〈任意〉; $\langle \text{InLane2}(A) \rangle$
L，M	匹配在车道1上持续加速一段较长时间的汽车。 $\langle \text{InLane1}(A) \rangle \wedge \langle \text{AvgAccelGt}(A) \rangle_{??}$ ∧〈持续时间不短〉
N	匹配汽车对：在汽车 A 转向的整个持续时间内，汽车 B 都位于车道2 从车道1转入车道2。 $\left(\langle \text{InLane1}(A) \rangle ; \langle \text{任意} \rangle ; \langle \text{InLane2}(A) \rangle \right) \wedge \langle \text{InLane2}(B) \rangle$
O，P	匹配位于车道1和车道2这两条平行车道上的汽车对，其中汽车 A 的速度比 汽车B持续较长一段时间。 $\langle \text{InLane1}(A) \rangle \wedge \langle \text{InLane2}(B) \rangle \wedge$〈持续时间不短〉$\wedge \langle$速度因子大于$(A, B) \rangle_{??}$
Q	匹配位于平行车道1和2中的汽车对，其中两辆汽车在 较长一段时间内彼此距离很近。 $\langle \text{InLane1}(A) \rangle \wedge \langle \text{InLane2}(B) \rangle \wedge$〈持续时间不短〉$\wedge \langle$距离小于$(A, B) \rangle_{??}$

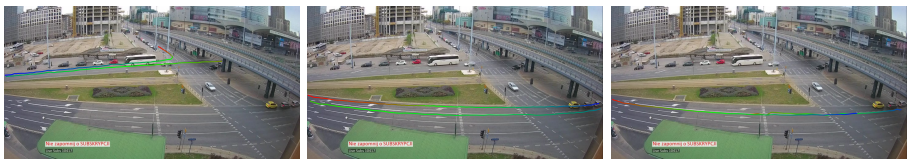


图6. 多目标查询选出的轨迹。每幅图展示两个对象；每个对象的颜色从红色渐变为绿色，以表示时间的推移。左：驶入有迎面来车车道的无保护右转。中：下方的汽车比上方的汽车开得快，并超过了它。右：一辆汽车紧跟在另一辆汽车后方行驶。

用于训练，后半部分的轨迹用于测试。我们从 Z_{train}中随机采样一组初始已标注示例 W ，其中2个样本为正类，10个为负类，然后从Z_{train}中主动标注另外25个示例。对于YTStreams和MABe22，标签来自真实查询。为使问题可处理，我们将搜索限制为最多包含三个谓词的草图，其中最多两个可以带有参数。在大多数情况下，这会将真值排除在搜索空间之外。

5.2 合成查询的准确性

我们展示了 QUIVR 仅凭少量带标签的示例就能合成准确的查询。我们评估了合成查询在 Z_{test}上的 F_1 分数。请注意，

表3. F_1 经过 n 步主动学习后的分数，采用我们的轨迹选择算法进行标注（“Q”）、主动学习消融实验（“R”）、长短期记忆网络（“L”）和 Transformer（“T”）。对于 Q 和 R，可能存在许多与已标注数据一致的查询，因此报告其中位数 F_1 分数。粗体表示给定步骤数下的最佳分数。

ID	0 步				5 步				10 步				25 步			
	Q	R	L	T	Q	R	L	T	Q	R	L	T	Q	R	L	T
A	0.69	0.69	**1.00**	0.74	1.00	**1.00**	1.00	0.74	**1.00**	1.00	1.00	0.74	**1.00**	1.00	1.00	0.74
B	**0.99**	**0.99**	0.47	0.20	**0.99**	**0.99**	0.47	0.25	**0.99**	**0.99**	0.47	0.05	**0.99**	0.98	0.47	0.06
C	**0.96**	**0.96**	0.38	0.09	**0.99**	0.96	0.38	0.08	**0.99**	0.96	0.38	0.02	**0.99**	0.98	0.38	0.01
D	**0.77**	**0.77**	0.52	0.27	**0.99**	0.96	0.52	0.28 <style 标识符="1">0.99		**0.99**	0.52	0.32	0.99	**1.00**	0.52	0.08
E	**1.00**	**1.00**	0.44	0.29	**1.00**	**1.00**	0.44	0.14	**1.00**	**1.00**	0.44	0.13	**1.00**	1.00 <style id="标识符">0.44		0.07
F	**0.88**	**0.88**	0.78	0.38	**0.99**	0.96	0.78	0.39	**1.00**	0.96	0.78	0.18	**1.00**	0.96	0.78	0.27
G	0.68	0.68	0.65 <style 标识符="1">0.78	**1.00**		**1.00**	0.65	0.77	**1.00**	**1.00**	0.65	0.77	**1.00**	**1.00**	0.65	0.77
H	**0.30**	**0.30**	0.12	0.22	0.34	**0.34**	0.13	0.23	**0.92**	**0.92**	0.13	0.22	**0.92**	**0.92**	0.13	0.37
I	**0.37**	**0.37**	0.13	0.00	**1.00**	0.37	0.13	0.00	**1.00**	**1.00**	0.13	0.00	**1.00**	**1.00**	0.13	0.31
J	0.07	0.07	0.01	**1.00**	0.41	0.07	0.01	**0.86**	**0.80**	0.09	0.04	0.75	0.80	0.09	0.04	**0.86**
K	**0.28**	**0.28**	0.15	0.00	**0.99**	0.27	0.15	0.00	**0.99**	**0.99**	0.15	0.00	**0.99**	**0.99**	0.15	0.00
L	**0.67**	**0.67**	0.07	0.37	**0.96**	0.88	0.07	0.42	**0.96**	0.88 <style 标识符="1">0.07	0.08 <style 标识符="1">0.96			0.88	0.07	0.31
M	**0.92**	**0.92**	0.10	0.37	**0.99**	0.92	0.10	0.46	**0.99**	0.92 <style 标识符="1">0.10	0.00 <style 标识符="1">0.99			0.92	0.10	0.18
N	**0.60**	**0.60**	0.02	0.00	**0.20**	0.09	0.02	0.00	0.11	**0.21**	0.02	0.00	0.18	**0.78**	0.02	0.31
O	**0.11**	**0.11**	0.01	0.04	**0.50**	0.17	0.01	0.21	**0.70**	0.17	0.01	0.21	**1.00**	0.21	0.01	0.00
P	**0.16**	**0.16**	0.04	0.04	**0.23**	0.21	0.03	0.04	**0.82**	0.21	0.03	0.14	**1.00**	0.29	0.03	0.14
Q	**0.07**	**0.07**	0.02	0.02	0.16	0.12	0.01	**0.31**	**0.92**	0.12	0.01	0.18	**1.00**	0.12	0.01	0.20

我们的算法返回一组 C 一致查询；我们报告了跨 $Q \in C$ 的中位数 F_1 分数。

基线方法。我们将其与以下方法进行比较：（i）消融实验：用一种方法取代我们的主动学习策略，从剩余未标注的训练样本中均匀随机标注 z；（ii）长短期记忆网络（LSTM） [33,16] 神经网络；以及（iii）Transformer神经网络 [77,29,26]。由于神经网络在如此小的数据集上表现不佳，我们在辅助任务（即轨迹预测） [43]上对 LSTM 进行预训练。随后，我们冻结已学习 LSTM 的隐藏表示，并将其用作特征，在已标注示例上训练逻辑回归模型。神经网络基线通过在未标注轨迹中选择预测为正类概率最高的轨迹来执行主动学习。

结果。 我们在表3中展示了主动学习经过 0、5、10 和 25 个步骤后 17 个查询各自的 F_1 得分。仅经过 10 个步骤，我们的方法就在 17 个查询中的 10 个上取得了高于 0.99 的F_1 得分；经过 25 个步骤后，除 2 个查询外，其余查询的 F_1 得分均高于 0.9。因此， QUIVR 能够以相对较少的用户输入合成准确的查询。神经网络的表现较差，尤其是在较难的查询上。

5.3 合成运行时间

接下来，我们表明定量剪枝和使用图形处理器各自都能显著缩短合成时间，并评估了主动学习25个步骤的总运行时间。

消融实验。 我们与两种消融实验进行比较：（i）使用 [53] 的二分搜索方法查找剪枝对，而非使用我们的定量语义；以及

表4。 在 CPU 和图形处理器上运行合成的运行时间（秒）（均值 ± 标准误），分别采用二分搜索（B）和定量语义（Q），并进行 25 个主动学习步骤。

ID	CPU		GPU	
	B	Q	B	Q
A	$8,460 \pm 1,517$	$3,343 \pm 202$	737 ± 36	174 ± 14
B	$3,511 \pm 549$	$2,291 \pm 237$	428 ± 37	110 ± 9
C	$3,319 \pm 505$	$2,007 \pm 359$	376 ± 6	113 ± 9
D	$2,728 \pm 476$	$2,714 \pm 334$	370 ± 8	119 ± 2
E	$1,264 \pm 176$	599 ± 54	225 ± 3	50 ± 1
F	$1,689 \pm 360$	748 ± 81	285 ± 7	60 ± 1
G	661 ± 141	133 ± 23	219 ± 77	30 ± 1
H	399 ± 70	147 ± 9	185 ± 94	32 ± 17
I	400 ± 74	84 ± 13	163 ± 85	23 ± 12
J	544 ± 120	173 ± 5	227 ± 121	36 ± 19
K	493 ± 77	125 ± 25	163 ± 83	30 ± 16
L	732 ± 47	286 ± 73	252 ± 133	57 ± 29
M	697 ± 40	253 ± 49	245 ± 128	56 ± 30
N	$5,691 \pm 272$	977 ± 176	$1,393 \pm 590$	264 ± 136
O	$8,306 \pm 521$	$2,314 \pm 476$	811 ± 12	127 ± 2
P	$11,326 \pm 673$	$4,198 \pm 1,333$	970 ± 60	167 ± 8
Q	$12,430 \pm 962$	$2,915 \pm 508$	$1,141 \pm 101$	183 ± 11

（ii）在 CPU 而非图形处理器上评估矩阵语义（附录 A.1）。

结果。 表4报告了我们的算法在 CPU（2×AMD EPYC 7402 24-Core）和图形处理器（1× NVIDIA RTX A6000）上的运行时间。对于二分搜索，图形处理器的平均速度比 CPU 快 $7.6\times$ 倍。在图形处理器上，相较于二分搜索，采用定量语义还能进一步提升 $5.0\times$ 倍速度。

6 相关工作

用于参数剪枝的单调性。我们的参数剪枝算法以 [49] 为基础。他们的方法已用于合成用于序列分类的STL公式：首先枚举草图，然后利用单调性寻找参数，这与我们的二分搜索基线方法 [53]类似。我们用一种基于定量语义的新策略取代二分搜索，使速度提升了 $5.0\times$。此外，还有研究基于 [49] ，通过计算参数满足区域（他们称之为“有效域”）之间的豪斯多夫距离，构建轨迹间与逻辑相关的距离度量，并将其应用于聚类 [78]。对于STL等逻辑，我们的定量语义可以提升他们方法的速度。

时序逻辑公式的合成。更广泛地说，已有研究通过离散化参数空间并沿满足边界 [24]逐步搜索，在一种 STL 变体中合成参数；在一维情况下，这种接近方式就变成了二分搜索，也因此继承了其缺点。也有研究合成能够被闭环控制模型 [38], 满足的 STL 公式，但这些方法假设能够为错误的 STL 公式找到反例轨迹，这不适用于我们的场景。另一种方法是使用基于梯度的优化 [35] 或随机优化，在 STL 公式中合成参数

[45], ，但我们发现这些方法在我们的场景中效果不佳，而且既没有可靠性保证，也没有完备性保证。还有研究使用决策树合成 STL 公式 [44,48,14,1], ，但这些方法仅适用于 STL 的一个受限子集，即由一组固定模板公式构成的布尔组合。这一限制使这些方法无法合成时序结构，而时序结构是我们领域中查询的关键组成部分。最后，也有研究使用决策树对 STL 公式进行主动学习 [48]„，但它假设能够查询某个特定 STL 公式与真值之间的等价性，而这在我们的场景中是不可能的。

合成常量。已有研究使用反例引导的归纳式合成以及不同的理论求解器来合成程序的参数，包括 Fourier–Motzkin变量消去和 SMT求解器 [2]。尽管我们的合成目标可以编码在线性算术理论中，但其规模极其庞大；我们发现，这类求解器在实践中并不奏效。

查询视频数据。近期已有研究针对视频数据中的目标检测和轨迹进行查询 [40,28,41,42,7,8,54,5]。主要区别在于，我们关注的是合成；此外，这些方法侧重于在预定义谓词上使用 select、inner-join、group-by 等类SQL算子，而这些算子无法表达我们语言中的顺序组合和迭代算子等组合形式；后者对于识别更复杂的行为是必需的。

神经符号模型。近期已有研究探索在机器学习背景下利用程序合成。例如，已有工作使用程序表示图像中的高层结构 [23,22,74,84,21], 、用于强化学习 [80,9,34,79], 以及查询网站 [18]；相比之下，我们使用程序对轨迹进行分类。与我们最密切相关的工作是合成作用于列表的函数式程序 [76,67]。我们的语言包含其语言中未提供的关键构造。最重要的是，我们包含顺序组合；在它们的函数式语言中，这种算子需要表示为一系列嵌套的 if-then-else算子。此外，它们的语言不支持匹配子序列的谓词；虽然可以添加这样的谓词，但它们的任何算子都无法组合这类谓词。

定量综合。已有研究探讨了具有定量属性的程序合成，例如，为近似计算 [65,57,37], 、概率编程 [52,15], 以及嵌入式控制 [56], 合成优化代码 [17]。这些方法大多侧重于基于搜索的综合，包括采用约束优化 [52], 、连续优化 [17],、枚举搜索 [15,57],或随机搜索 [65,56,37]。尽管我们借鉴了这一领域的研究思路，但我们基于定量语义的剪枝策略是全新的。

定量语义。 我们的定量语义与时序逻辑公式的“鲁棒度” [25] 相似。区别在于，通过调整基本谓词的指称，我们的定量语义能够为满足边界上的参数赋值。更广泛地说，已有研究探讨了定量

时序逻辑语义在鲁棒约束满足 [25,73,20], 中的应用，以及如何借此指导强化学习 [39]。也有研究探讨了定量正则表达式（QRE） [4], ，但由于其存在非确定性，通常无法高效求值；而我们的语言经过限制，以确保计算高效。另有研究利用二分搜索计算决策阈值，以合成用于网络流量分类的 QRE [68], 。类似地，有研究使用维特比半环为 Datalog程序 [69], 获取定量语义，并结合梯度下降学习 Datalog程序的规则。相比之下，我们利用定量语义，以可证明正确的方式高效剪枝参数搜索空间。最后，也有研究使用 GPU 对正则表达式 [55]求值；不过，他们关注的是字符串上的正则表达式。

查询语言。 我们的语言与信号时序逻辑（STL） [50]和带测试的克林代数（KAT） [46]都密切相关。具体而言，它可以直接扩展以涵盖这两者（详情请参见附录 A），并且我们的剪枝策略适用于这一扩展后的语言。为涵盖 KAT 而引入的克莱尼星号会增加评估时间。STL 已被用于监控自动驾驶车辆的安全需求 [32]。时空感知逻辑（SPTL）是 STL 的扩展，用于支持空间推理 [31]。其中许多运算符是单调的，因此可以从我们的算法中受益。Scenic [27] 是一种用于创建静态和动态驾驶场景的领域特定语言，但其重点在于生成场景，而不是查询行为。

7 结论

我们提出了一种名为 Quivr 的新型框架，用于合成针对视频轨迹数据的查询。我们的语言类似于 KAT 和 STL，但支持对多步谓词进行合取和顺序组合。仅给出少量示例， Quivr 即可高效地合成与这些示例一致的轨迹查询。我们方法的一项重要贡献是使用定量语义来剪枝参数搜索空间，相较于当前最先进的方法实现了 $5.0\times$ 的加速。在评估中，我们证明了 Quivr 能够有效地合成查询，以识别有趣的驾驶行为，并且我们的优化显著缩短了合成时间。

致谢 感谢匿名审稿人提出的宝贵意见。本研究部分由美国国家科学基金会资助项目 CCF-1910769、美国国家科学基金会资助项目 CCF-1917852 和美国陆军研究办公室资助项目 W911NF-20-1-0080 支持。

数据可用性声明 我们提供了一个研究制品 [51] ，其中包含我们的实现，可复现我们的实验结果（表3和表4），也可用于在其他轨迹数据集上进行合成，或为其他领域特定语言实现我们的算法。

参考文献

1. Aasi, E., Vasile, C.I., Bahreinian, M., Belta, C.：使用提升决策树对时间序列数据进行分类。载于：2022年IEEE/RSJ智能机器人与系统国际会议（IROS）。第1263–1268页（2022年）。https://doi.org/10.1109/IROS47612.2022.9982105 2. Abate, A., David, C., Kesseli, P., Kroening, D., Polgreen, E.：基于反例引导的模理论归纳合成。载于：Chockler, H., Weissenbacher, G.（编），计算机辅助验证。第270–288页。Springer International Publishing，Cham（2018年）3. Alur, R., Bodik, R., Juniwal, G., Martin, M.M., Raghothaman, M., Seshia, S.A., Singh, R., Solar-Lezama, A., Torlak, E., Udupa, A.：语法引导合成。IEEE（2013年）4. Alur, R., Mamouras, K., Ulus, D.：定量正则表达式的导数。载于：模型、算法、逻辑与工具，第75–95页。Springer（2017年）5. Bastani, F., Bastani, O., Balasingam, A., He, S., Jiang, Z., Mittal, R., Alizadeh, M., Balakrishnan, H., Kraska, T., Madden, S.：SkyQuery：优化无人机上的视频查询（2019年）6. Bastani, F., Bastani, O., Balasingam, A., He, S., Jiang, Z., Mittal, R., Alizadeh, M., Balakrishnan, H., Kraska, T., Madden, S.：SkyQuery：优化无人机上的视频查询。载于：Onward!（2021年）7. Bastani, F., He, S., Balasingam, A., Gopalakrishnan, K., Alizadeh, M., Balakrishnan, H., Cafarella, M., Kraska, T., Madden, S.：MIRIS：快速视频目标跟踪查询。载于：2020年ACM SIGMOD数据管理国际会议论文集。第1907–1921页（2020年）8. Bastani, F., Moll, O., Madden, S.：Vaas：大规模视频分析。《VLDB基金会会刊》**13**(12)，2877–2880（2020年）9. Bastani, O., Pu, Y., Solar-Lezama, A.：通过策略提取实现可验证的强化学习。载于：神经信息处理系统进展。第2494–2504页（2018年）10. Bergmann, P., Meinhardt, T., Leal-Taixe, L.：不靠花哨技巧的跟踪。载于：IEEE计算机视觉国际会议论文集。第941–951页（2019年）11. Bertinetto, L., Valmadre, J., Henriques, J.F., Vedaldi, A., Torr, P.H.：用于目标跟踪的全卷积孪生网络。载于：欧洲计算机视觉会议。第850–865页。Springer（2016年）12. Betke, M., Hirsh, D.E., Bagchi, A., Hristov, N.I., Makris, N.C., Kunz, T.H.：在杂波中跟踪数量众多且不断变化的对象。载于：2007年IEEE计算机视觉与模式识别会议。第1–8页。IEEE（2007年）13. Bock, J., Krajewski, R., Moers, T., Runde, S., Vater, L., Eckstein, L.：IND数据集：德国交叉路口自然场景道路使用者轨迹无人机数据集。arXiv预印本 arXiv:1911.07602（2019年）14. Bombara, G., Belta, C.：使用决策树离线和在线学习信号时序逻辑公式。《ACM网络物理系统汇刊》**5**(3)（2021年3月）。https://doi.org/10.1145/3433994, https://doi.org/10.1145/343399415. Bornholt, J., Torlak, E., Grossman, D., Ceze, L.：利用元草图优化合成。载于：第43届ACM SIGPLAN-SIGACT编程语言原理年会论文集。第775–788页（2016年）

16. Chang, M.F., Lambert, J.W., Sangkloy, P., Singh, J., Bak, S., Hartnett, A., Wang, D., Carr, P., Lucey, S., Ramanan, D., Hays, J.：Argoverse：基于丰富地图的三维跟踪与预测。载于：计算机视觉与模式识别会议（CVPR）（2019年）17. Chaudhuri, S., Clochard, M., Solar-Lezama, A.：利用平滑证明搜索连接布尔综合与定量综合。载于：第41届 ACM SIGPLAN-SIGACT 编程语言原理研讨会论文集，第207–220页（2014年）18. Chen, Q., Lamoreaux, A., Wang, X., Durrett, G., Bastani, O., Dillig, I.：利用神经符号程序合成实现网络问答。载于：第42届ACM SIGPLAN编程语言设计与实现国际会议论文集，第328–343页（2021年）19. Dasgupta, S.：贪心主动学习策略分析。载于：神经信息处理系统进展，第337–344页（2005年）20. Deshmukh, J.V., Donz´e, A., Ghosh, S., Jin, X., Juniwal, G., Seshia, S.A.：信号时序逻辑的鲁棒在线监控。《系统设计中的形式化方法》**51**(1), 5–30（2017年）21. Ellis, K., Nye, M., Pu, Y., Sosa, F., Tenenbaum, J., Solar-Lezama, A.：编写、执行、评估：基于交互式解释器的程序合成。载于：神经信息处理系统进展，第9169–9178页（2019年）22. Ellis, K., Ritchie, D., Solar-Lezama, A., Tenenbaum, J.：学习从手绘图像推断图形程序。载于：神经信息处理系统进展，第6059–6068页（2018年）23. Ellis, K., Solar-Lezama, A., Tenenbaum, J.：通过程序合成进行无监督学习。载于：神经信息处理系统进展，第973–981页（2015年）24. Ergurtuna, M., Gol, E.A.：一种基于过去时信号时序逻辑的高效公式综合方法。IFAC-PapersOnLine**52**(11), 43–48（2019年）。https://doi.org/https://doi.org/10.1016/j.ifacol.2019.09.116，https://www.sciencedirect.com/science/article/pii/S2405896319307451，第5届IFAC智能控制与自动化科学会议（ICONS 2019）25. Fainekos, G.E., Pappas, G.J.：连续时间信号的时序逻辑规范鲁棒性。理论计算机科学**410**(42), 4262–4291（2009年）26. Franco, L., Placidi, L., Giuliari, F., Hasan, I., Cristani, M., Galasso, F.：深入探究用于轨迹预测的变换器网络。《模式识别》**138**，109372 2023 10 1016 2023 109372，（）。https://doi.org/https://doi.org/ . /j.patcog. . https://www.sciencedirect.com/science/article/pii/S0031320323000730 27. Fre mont, D.J., Dreossi, T., Ghosh, S., Yue, X., Sangiovanni-Vincentelli, A.L., Se- shia, S.A.：Scenic：用于场景规范和场景生成的语言。载于：第40届ACM SIGPLAN编程语言设计与实现会议论文集，第63–78页（2019年）28. Fu, D.Y., Crichton, W., Hong, J., Yao, X., Zhang, H., Truong, A., Narayan, A., Agrawala, M., R´e, C., Fatahalian, K.：Rekall：通过时空标签的组合来指定视频事件。arXiv预印本 arXiv:1910.02993（2019年）29. Giuliari, F., Hasan, I., Cristani, M., Galasso, F.：用于轨迹预测的变换器网络（2020年）30. He, K., Zhang, X., Ren, S., Sun, J.：用于图像识别的深度残差学习。载于：IEEE计算机视觉与模式识别会议论文集，第770–778页（2016年）31. Hekmatnejad, M., Hoxha, B., Deshmukh, J.V., Yang, Y., Fainekos, G.：利用时空感知逻辑形式化并评估自动驾驶车辆感知系统的需求（2022年）

32. Hekmatnejad, M., Yaghoubi, S., Dokhanchi, A., Amor, H.B., Shrivastava, A., Karam, L., Fainekos, G.：在信号时序逻辑中编码并监控自动驾驶车辆的责任敏感安全规则。载于：第17届 ACM-IEEE 系统设计形式化方法与模型国际会议论文集。MEMOCODE '19，美国计算机协会，纽约，NY，美国（2019年）。https://doi.org/10.1145/3359986.3361203，https://doi.org/10.1145/3359986.336120333. Hochreiter, S., Schmidhuber, J.：长短期记忆。《神经计算》**9**(8), 1735–1780（1997年）34. Inala, J.P., Bastani, O., Tavares, Z., Solar-Lezama, A.：合成具有归纳泛化能力的程序化策略。载于：国际学习表征会议（2019年）35. Jha, S., Tiwari, A., Seshia, S.A., Sahai, T., Shankar, N.：Telex：使用紧致度指标从正例中学习信号时序逻辑。《系统设计中的形式化方法》**54**(3), 364–387（2019年11月）。https://doi.org/10.1007/s10703-019- 003321 36 1, 101007107030190033 2-https://doi.org/./s---. Ji, R., Liang, J., Xiong, Y., Zhang, L., Hu, Z.：交互式程序合成中的问题选择。载于：第41届ACM SIGPLAN编程语言设计与实现会议论文集，第1143–1158页（2020年）37. Jia, Z., Thomas, J., Warszawski, T., Gao, M., Zaharia, M., Aiken, A.：利用松弛图替换优化深度神经网络计算。载于：第2届系统与机器学习会议（SysML'19）（2019年）38. Jin, X., Donz´e, A., Deshmukh, J. V., Seshia, S.A.：从闭环控制模型中挖掘需求。IEEE《计算机辅助集成电路与系统设计学报》**34**(11), 1704–1717（2015年）。https://doi.org/10 2421907 39 1109 2015 . / TCAD. . . Jothimurugan, K., Alur, R., Bastani, O.：一种用于强化学习任务的可组合规范语言。载于：神经信息处理系统进展，第13041–13051页（2019年）40. Kang, D., Bailis, P., Zaharia, M.：Blazeit：优化基于神经网络的视频分析中的声明式聚合和限制查询。《VLDB基金会会刊》**13**(4), 533–546（2019年）41. Kang, D., Mathur, A., Veeramacheneni, T., Bailis, P., Zaharia, M.：联合优化基于深度神经网络的视觉分析中的预处理和推理。arXiv预印本 arXiv:2007.13005（2020年）42. Kang, D., Raghavan, D., Bailis, P., Zaharia, M.：用于监控和改进机器学习模型的模型断言。arXiv预印本 arXiv:2003.01668（2020年）43. Kitani, K.M., Ziebart, B.D., Bagnell, J.A., Hebert, M.：活动预测。载于：欧洲计算机视觉会议，第201–214页。Springer（2012年）44. Kong, Z., Jones, A., Belta, C.：用于学习和检测异常行为的时序逻辑。IEEE《自动控制学报》**62**(3), 1210–1222（2017年）。https://doi.org/10 2585083 45 1109 2016 . /TAC. . . Kong, Z., Jones, A., Medina Ayala, A., Aydin Gol, E., Belta, C.：从数据中推断用于分类和预测的时序逻辑。载于：第17届混合系统：计算与控制国际会议论文集，第273–282页。HSCC '14，美国计算机协会，纽约，NY，美国（2014年）。https://doi.org/10.1145/2562059.2562146，https://doi.org/10.1145/2562059.256214646. Kozen, D.：带测试的克莱尼代数。ACM《编程语言与系统汇刊》（TOPLAS）**19**(3), 427–443（1997年）

47. Krizhevsky, A., Sutskever, I., Hinton, G.E.：使用深度卷积神经网络进行ImageNet 分类。《ACM 通讯》**60**(6), 84–90（2017年）48. Linard, A., Tumova, J.：信号时序逻辑规范的主动学习。载于：2020年 IEEE 第16届自动化科学与工程国际会议（CASE）。第779–785页（2020年）。https://doi.org/10.1109/CASE.2020.9216778 49. Maler, O.：偏序域的单调划分学习（进行中的工作）（2017年7月），https://hal.science/hal-01556243，工作论文或预印本 50. Maler, O., Nickovic, D.：连续信号时序属性的监控。载于：Lakhnech, Y., Yovine, S.（编）《定时与容错系统的形式化技术、建模与分析》。第152–166页。施普林格·柏林·海德堡，柏林，海德堡（2004年）51. Mell, S.：从示例合成轨迹查询的研究制品（2023年5月）。https://doi.org/10.5281/zenodo.7931441 52. Misailovic, S., Carbin, M., Achour, S., Qi, Z., Rinard, M.C.：Chisel：近似计算内核的可靠性与精度感知优化。《ACM SIGPLAN 通知》**49**(10), 309–328（2014年）53. Mohammadinejad, S., Deshmukh, J.V., Puranic, A.G., Vazquez-Chanlatte, M., Donz´e, A.：采用高效枚举技术对时间序列数据进行可解释分类。载于：第23届混合系统：计算与控制国际会议论文集。HSCC '20，美国纽约：计算机协会（2020年）。https://doi.org/10.1145/3365365.3382218，https://doi.org/103382218 5411453365365./.. Moll, O., Bastani, F., Madden, S., Stonebraker, M., Gadepally, V., Kraska, T.：Exsample：通过自适应采样高效搜索视频库。arXiv 预印本 arXiv:2005.09141（2020年）55. Naghmouchi, J., Scarpazza, D.P., Berekovic, M.：通用图形处理器上的小规则集正则表达式匹配：定量性能分析与优化。载于：第24届 ACM 超级计算国际会议论文集。第337–348页（2010年）56. Nori, A.V., Ozair, S., Rajamani, S.K., Vijaykeerthy, D.：概率程序的高效合成。《ACM SIGPLAN 通知》**50**(6), 208–217（2015年）57. Phothilimthana, P.M., Thakur, A., Bodik, R., Dhurjati, D.：扩展超优化规模。载于：第二十一届编程语言与操作系统体系结构支持国际会议论文集。第297–310页（2016年）58. Preiss, J.A., Honig, W., Sukhatme, G.S., Ayanian, N.：Crazyswarm：大型纳米四旋翼无人机集群。载于：2017年 IEEE机器人与自动化国际会议（ICRA）。第3299–3304页。IEEE（2017年）59. Redmon, J., Farhadi, A.：YOLOv3：渐进式改进。arXiv 预印本 arXiv:1804.02767（2018年）60. Ren, S., He, K., Girshick, R., Sun, J.：Faster R-CNN：利用区域建议网络实现实时目标检测。载于：神经信息处理系统进展。第91–99页（2015年）61. Robicquet, A., Sadeghian, A., Alahi, A., Savarese, S.：学习社交礼仪：拥挤场景中的人类轨迹预测。载于：欧洲计算机视觉会议（ECCV）（2016年）62. Roy, N., McCallum, A.：通过误差降低的采样估计实现最优主动学习。机器学习国际会议（2001年）63. Sadigh, D., Sastry, S.S., Seshia, S.A., Dragan, A.：针对人类内部状态的信息收集行为。载于：2016年 IEEE/RSJ 智能机器人与系统国际会议（IROS）。第66–73页。IEEE（2016年）

64. Sadigh, D., Sastry, S., Seshia, S.A., Dragan, A.D.：规划能够利用对人类行为影响的自动驾驶汽车。载于：《机器人：科学与系统》。第2卷。美国密歇根州安娜堡（2016年）65. Schkufza, E., Sharma, R., Aiken, A.：随机超优化。《ACM SIGARCH计算机体系结构新闻》**41**(1), 305–316（2013年）66. Schmerling, E., Leung, K., Vollprecht, W., Pavone, M.：用于人机交互的多模态概率模型规划。载于：2018年 IEEE机器人与自动化国际会议（ICRA）。第1–9页。IEEE（2018年）67. Shah, A., Zhan, E., Sun, J., Verma, A., Yue, Y., Chaudhuri, S.：使用可采纳神经启发式学习可微程序。神经信息处理系统进展**33**（2020年）68. Shi, L., Li, Y., Loo, B.T., Alur, R.：通过程序合成进行网络流量分类。载于：Groote, J.F., Larsen, K.G.（编）《系统构造与分析工具和算法》。第430–448页。施普林格国际出版公司，Cham（2021年）69. Si, X., Raghothaman, M., Heo, K., Naik, M.：使用数值松弛合成 Datalog程序。载于：IJCAI（2019年）70. Silver, D., Huang, A., Maddison, C.J., Guez, A., Sifre, L., Van Den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., 等：利用深度神经网络和树搜索掌握围棋。《自然》**529**(7587), 484–489（2016年）71. Solar-Lezama, A.：基于草图的程序合成。加州大学伯克利分校（2008年）72. Sun, J.J., Geuther, B., Kumar, V., Robie, A., Schretter, C., Sheppard, K., Chakraborty, D., Branson, K., Kennedy, A.：用于多智能体行为表征学习的 MABe22 基准（2022年6月）。https://doi.org/10.20186/D.73.22002 1. Tabuada, P., Neider, D.：鲁棒线性时序逻辑。《计算机科学逻辑 2016》（2016年）74. Tian, Y., Luo, A., Sun, X., Ellis, K., Freeman, W.T., Tenenbaum, J.B., Wu, J.：学习推断和执行三维形状程序。arXiv 预印本 arXiv:1901.02875（2019年）75. Tweed, D., Calway, A.：在野生动物影像中追踪多只动物。载于：《用户交互辅助服务机器人目标识别》。第2卷，第24–27页。IEEE（2002年）76. Valkov, L., Chaudhari, D., Srivastava, A., Sutton, C., Chaudhuri, S.：Houdini：将终身学习作为程序合成。载于：神经信息处理系统进展。第8687–8698页（2018年）77. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L.u., Polosukhin, I.：注意力就是一切。载于：Guyon, I., Luxburg, U.V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., Garnett, R.（编）《神经信息处理系统进展》。第30卷。Curran Associates, Inc.（2017年），https://proceedings.neurips.cc/paper files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf 78. Vazquez-Chanlatte, M., Ghosh, S., Deshmukh, J.V., Sangiovanni-Vincentelli, A., Seshia, S.A.：利用单调逻辑性质进行时间序列学习。载于：Colombo, C., Leucker, M.（编）《运行时验证》。第389–405页。施普林格国际出版公司，Cham（2018年）79. Verma, A., Le, H., Yue, Y., Chaudhuri, S.：模仿投影式程序化强化学习。载于：神经信息处理系统进展。第15752–15763页（2019年）

80. Verma, A., Murali, V., Singh, R., Kohli, P., Chaudhuri, S.：以程序化方式可解释的强化学习。载于：ICML（2018年）81. Weinstein, A., Cho, A., Loianno, G., Kumar, V.：视觉惯性里程计集群：由基于视觉的四旋翼无人机构成的自主集群。《IEEE机器人与自动化快报》**3**(3)，1801–1807（2018年）82. Wishart, J., Como, S., Elli, M., Russo, B., Weast, J., Altekar, N., James, E., Chen, Y.：配备高级驾驶辅助系统的车辆的驾驶安全性能评估指标。SAE技术论文 **2**(2020-01-1206)（2020年）83. Wojke, N., Bewley, A., Paulus, D.：使用深度关联度量进行简单的在线实时跟踪。载于：2017年IEEE国际图像处理会议（ICIP），第3645–3649页。IEEE（2017年）84. Young, H., Bastani, O., Naik, M.：通过程序合成学习神经符号生成模型。载于：ICML（2019年）85. Zhan, E., Tseng, A., Yue, Y., Swaminathan, A., Hausknecht, M.：使用程序化风格一致性学习可校准策略。载于：ICML（2020年）

$$[\![\varphi_{??}]\!]^M(z) \coloneqq \begin{bmatrix} \iota_\varphi(z_{0:0}) & \iota_\varphi(z_{0:1}) & \cdots & \iota_\varphi(z_{0:n}) \\ -\infty & \iota_\varphi(z_{1:1}) & \cdots & \iota_\varphi(z_{1:n}) \\ \vdots & \vdots & \ddots & \vdots \\ -\infty & -\infty & \cdots & \iota_\varphi(z_{n:n}) \end{bmatrix}$$

$$[\![Q_1 \wedge Q_2]\!]^M(z) \coloneqq \min\{[\![Q_1]\!]^M(z), [\![Q_2]\!]^M(z)\}$$

$$[\![Q_1 \,;\, Q_2]\!]^M(z) \coloneqq [\![Q_1]\!]^M(z) \cdot [\![Q_2]\!]^M(z)$$

图 7. 我们语言的矩阵语义； $z \in \mathcal{Z}$是一条长度为 n 的轨迹；min 表示逐元素取最小值；$\cdot$是将加法替换为取最大值、将乘法替换为取最小值后的矩阵乘积。我们省略 $[\![\varphi_\theta]\!]^M(z)$ 的情况，它与 $[\![\varphi_{??}]\!]^M(z)$ 相同，只需将 $\iota_\varphi(z_{i:j})$ 替换为 $[\![\varphi]\!]^q(z_{i:j})$。

A. 查询语言扩展

A.1 使用矩阵计算定量语义

序列运算符要求考虑轨迹的任意切分，这可以通过动态规划实现。然而，这项评估高度适合数据并行处理，而这正是 GPU 的优势所在。由于矩阵运算天然适合 GPU 编程，我们为该语言设计了矩阵语义，从而进一步显著提升了速度。

我们的想法是不仅计算轨迹的定量语义，还同时计算每个子序列的语义，即$[\![Q]\!]^M_{v,\,u}(z)$ 是一个 $(n+1) \times (n+1)$ 矩阵，其中第 $i,\ j$ 个条目给出从 i 到 j 的子序列的定量语义——也就是说，我们定义语义，使得下面的定理 4 成立。

我们在图 7 中定义矩阵语义。基本情况 $\varphi_{??}$ evaluates$^{\iota_\varphi}$ 作用于每个子序列，合取是逐元素取最小值，而顺序组合则是在最大值-最小值半环中进行矩阵乘法。直接采用标准矩阵乘法技术进行计算，需要 $O(n^3)$ 的时间和空间。然而，由于我们最终感兴趣的是

$$[\![Q]\!]^q_{v,u}(z) = \big([\![Q]\!]^M_{v,u}(z)\big)_{0,n} = \big[\infty, -\infty, \ldots, -\infty\big] \cdot \big([\![Q]\!]^M_{v,u}(z)\big) \cdot \big[-\infty, \ldots, -\infty, \infty\big]^T$$

我们可以重新结合矩阵乘积，将其转化为矩阵-向量乘积。这只需要 $O(n^2)$ 的时间和空间，与动态规划相同。

引理 3. 对于任意 Q、向量 $v \in \mathbb{R}^d$、正向量 $u \in \mathbb{R}^d_{>0}$ 以及长度为 n 的轨迹z，$[\![Q]\!]^M_{v,\,u}(z)$ 是上三角矩阵——即，$([\![Q]\!]^M_{v,\,u}(z))_{i,\,j} = -\infty$ 对所有$i > j$均成立。

证明。 对 Q 进行归纳。根据定义，基础情形是上三角的；逐点最小值和最大-最小矩阵乘法也都保持上三角性。

定理 4。对于任意查询 Q、向量 $v \in \mathbb{R}^d$、正向量 $u \in \mathbb{R}^d_{>0}$、长度为 n 的轨迹 z 以及索引 $i \leq j$，$(\llbracket Q \rrbracket^M_{v,u}(z))_{i,j} = \llbracket Q \rrbracket^q_{v,u}(z_{i:j})$。特别地，$\llbracket Q \rrbracket^q_{v,u}(z) = (\llbracket Q \rrbracket^M_{v,u}(z))_{0,n}$。

证明。我们对 Q 进行结构归纳。基础情形 $Q = \varphi$ 和 $Q = \varphi_{??}$ 由 $\llbracket \varphi_{??} \rrbracket^q(z_{i:j})$ 和 $(\llbracket \varphi_{??} \rrbracket^M(z))_{i,j}$ 的定义可得。

对于情形 $Q = Q_1 ; Q_2$，我们有

$$\llbracket Q_1 ; Q_2 \rrbracket^M_{v,u}(z)_{i,j} = \max_{1 \leq k \leq n} \min\{\llbracket Q_1 \rrbracket^M_{v,u}(z)_{i,k}, \llbracket Q_2 \rrbracket^M_{v,u}(z)_{k,j}\}$$

但由于 $\llbracket Q_1 \rrbracket^M_{v,u}(z)$ 和 $\llbracket Q_2 \rrbracket^M_{v,u}(z)$ 是上三角的，因此我们有

$$= \max_{i \leq k \leq j} \min\{\llbracket Q_1 \rrbracket^M_{v,u}(z)_{i,k}, \llbracket Q_2 \rrbracket^M_{v,u}(z)_{k,j}\}$$

最后，根据归纳假设，我们有

$$\begin{aligned} &= \max_{k \in \{i, i+1, \ldots, j\}} \min\{\llbracket Q_1 \rrbracket^q_{v,u}(z_{i:k}), \llbracket Q_2 \rrbracket^q_{v,u}(z_{k:j})\} \\ &= \llbracket Q_1 ; Q_2 \rrbracket^q_{v,u}(z_{i:j}). \end{aligned}$$

最后，对于情形 $Q = Q_1 \wedge Q_2$，我们有

$$\begin{aligned} \llbracket Q_1 \wedge Q_2 \rrbracket^M_{v,u}(z)_{i,j} &= \min\{\llbracket Q_1 \rrbracket^M_{v,u}(z)_{i,j}, \llbracket Q_2 \rrbracket^M_{v,u}(z)_{i,j}\} \\ &= \min\{\llbracket Q_1 \rrbracket^q_{v,u}(z_{i:j}), \llbracket Q_2 \rrbracket^q_{v,u}(z_{i:j})\} \\ &= \llbracket Q_1 \wedge Q_2 \rrbracket^q_{v,u}(z_{i:j}). \end{aligned}$$

A.2 附加算子

我们可以为语言添加几个附加算子，包括析取和克莱尼星号。克莱尼星号使我们能够涵盖正则表达式；此外，否定和另一个算子则使我们能够涵盖信号时序逻辑。

$$Q ::= \ldots \mid Q \vee Q \mid Q^* \mid \neg Q \mid Q^{\dashv [a,b]}$$

析取的语义很直观，与合取相同，只是使用最大值而不是最小值。

$$\llbracket Q_1 \vee Q_2 \rrbracket^M_{v,u}(z) \coloneqq \max\{\llbracket Q_1 \rrbracket^M_{v,u}(z), \llbracket Q_2 \rrbracket^M_{v,u}(z)\}$$

克莱尼星号是对重复顺序组合进行析取。与其他算子相比，克莱尼星号的评估渐近复杂度关于 n 更高。

$$\llbracket Q^* \rrbracket^M_{v,u}(z) \coloneqq \max_{0 \leq k \leq n} (\llbracket Q \rrbracket^M_{v,u}(z))^k$$

对于否定，请注意

$$\neg \llbracket \varphi_\theta \rrbracket(z) = \neg \mathbb{1}(\iota_\varphi(z) \geq \theta) = \mathbb{1}(\iota_\varphi(z) < \theta) = \mathbb{1}(-\iota_\varphi(z) > -\theta)$$

因此，如果我们为每个原子公式 φ 引入一个否定形式 $\bar{\varphi}$ where$\iota_{\bar{\varphi}}(z) := -\iota_{\varphi}(z)$, 那么我们有

$$\mathbb{1}(-\iota_{\varphi}(z) > -\theta) \approx \mathbb{1}(-\iota_{\varphi}(z) \geq -\theta) = [\![\bar{\varphi}_{-\theta}]\!](z).$$

在此我们忽略位于满足区域边界上的点所涉及的技术细节，以及这些点是否属于满足区域本身（即当$-\iota_{\varphi}(z) = \theta$时）。我们将 Q 定义为与 Q相同的公式，区别仅在于将每个原子谓词φ 替换为 $\bar{\varphi}$。然后，我们可以将否定的语义定义如下：

$$\left([\![\neg Q]\!]^M_{v,u}(z)\right)_{i,j} := \begin{cases} -\left([\![\bar{Q}]\!]^M_{v,u}(z)\right)_{i,j} & i \leq j \\ -\infty \end{cases}$$

运算符$(\cdot)^{\dashv[a,b]}$ 的存在是为了支持信号时序逻辑的“直到”运算符。为定义其语义，我们使用两个辅助函数。第一个函数接受一个向量，该向量表示信号在每个时间步上的值，并将其转换为一个矩阵，该矩阵表示信号在介于 a与 b 个时间步之间的每个区间上的最小值：

$$(\cdot)^{[a,b]} : \mathbb{R}^n \to \mathbb{R}^{n \times n}$$
$$(v^{[a,b]})_{i,j} := \begin{cases} \min\limits_{i \leq k \leq j} v_k & a \leq j - i \leq b \\ -\infty \end{cases}$$

第二个函数接受每个时间区间对应的值矩阵，并返回一个向量：对于每个时间步 i, 该向量给出从 i 到序列末尾这一区间的值：

$$(\cdot)^{\dashv} : \mathbb{R}^{n \times n} \to \mathbb{R}^n$$
$$(M^{\dashv})_i := M_{i,n}$$

现在我们可以定义该运算符的语义：

$$[\![Q^{\dashv[a,b]}]\!]^M_{v,u}(z) := ([\![Q]\!]^M_{v,u}(z)^{\dashv})^{[a,b]}$$

最后，我们可以定义从 STL公式 的翻译 $(\!|\cdot|\!)$ 。这个过程很直接，只有“直到”运算符例外，它会分解为顺序组合和 $(\cdot)^{\dashv[a,b]}$ 运算符：

$$(\!|\varphi|\!) := \varphi$$
$$(\!|\neg Q_1|\!) := \neg(\!|Q|\!)$$
$$(\!|Q_1 \wedge Q_2|\!) := (\!|Q_1|\!) \wedge (\!|Q_2|\!)$$
$$(\!|Q_1 \mathbf{U}_{[a,b]} Q_2|\!) := (\!|Q_1|\!)^{\dashv[a,b]} ; (\!|Q_2|\!)$$

B 数据集的谓词和真值查询

在所有数据集上，我们都会隐式包含 ⟨任意⟩ 和 ⟨无⟩ 谓词，分别匹配任意长度的所有轨迹和不匹配任何轨迹。

表5. MABe22 数据集使用的谓词。

谓词	描述
距离小于$_\theta$	在该区间的每个时间步，两者之间的距离是否对象少于 θ。
DistanceGt$_\theta$	在该区间的每个时间步，两者之间的距离是否对象多于 θ。
奔跑(A)	在该区间的每个时间步，小鼠 A 的速度是否处于至少 15 厘米/秒。
缓慢移动(A)	在该区间的每个时间步，小鼠 A 的速度是否至多为 2 厘米/秒。
持续时间短	该区间是否最长为 5 秒。
持续时间不短	该区间的持续时间是否至少为 5 秒。
持续时间长	该区间的持续时间是否至少为 10 秒。
鼻鼻接触(A,B)	在该区间的每个时间步，小鼠之间的距离是否小鼠 A 的鼻子与小鼠 B 的鼻子之间的距离至多为 1.5 厘米。
鼻-生殖器接触(A,B)	在该区间的每个时间步，小鼠之间的距离是否小鼠 A 的鼻子与小鼠 B 的尾根之间的距离至多为 1.5 厘米。
鼻-耳接触(A,B)	在该区间的每个时间步，小鼠之间的距离是否A 的鼻子与小鼠 B 的一只耳朵相距不超过 1.5 厘米。
朝向移动(A,B)	是否在该区间的每个时间步，A 的移动角度都接近 A 与 B 之间的角度。
凝视(A,B)	是否在该区间的每个时间步，A 的头部都朝向B.

YTStreams 数据集中有三类查询：涩谷视频中的单对象查询、华沙视频中的单对象查询，以及华沙视频中的双对象查询。涩谷和华沙的车道数量不同；对于单对象查询，某些谓词不适用。计入这些数量差异后，涩谷有 9 个谓词，华沙单对象查询有 10 个，华沙双对象查询有 19 个。计入数量差异后，MABe22 有 16 个谓词。海上监视任务有 8 个谓词。

查询的评估方式在海上监视任务与其他任务之间略有不同。在海上监视任务中，若 $[\![Q]\!](z) = 1$，则认为轨迹 z 与查询 Q 匹配。其他任务取自视频片段，因此可能无法表示完整轨迹（在我们的数据集中，轨迹可能出现在道路中段，而不是从侧面进入）。在这些情况下，若存在某个子轨迹与查询匹配，我们就将该轨迹视为正类，因此，对于某个 $i \leq j$，$[\![Q]\!](z_{i:j}) = 1$。这等价于使用 $\langle \text{Any} \rangle$ 对 Q 的起点和终点进行顺序组合，因此检查 $[\![\langle \text{Any} \rangle ; Q ; \langle \text{Any} \rangle]\!](z) = 1$。

YTStreams 的谓词见表1，YTStreams 的真值查询见表2。MABe22 的谓词见表5，MABe22 的真值查询见表7。海上监视的谓词见表6。

表6. 海上监视任务中使用的谓词。

谓词	描述
持续时间大于$_\theta$	该区间的持续时间是否至少为 θ 秒。
持续时间小于$_\theta$	区间跨度是否不超过 θ 秒。
X位置大于$_\theta$	区间内的每个时间步，X位置是否都至少为 θ。
X位置小于$_\theta$	区间内的每个时间步，X位置是否都至多为 θ。
Y位置大于$_\theta$	在该区间的每个时间步，Y位置是否至少为 θ。
Y位置小于$_\theta$	在该区间的每个时间步，Y位置是否至多为 θ。

表7。 MABe22任务的真值查询。“ID”表示其他表中评估的哪些查询属于这些通用查询的实例。这些查询基于MABe22 数据集中的行为标注 [72]。（参见其附录B.1中的表2。）

IDs	查询
A	**接近**：小鼠从至少相距 5 厘米的位置移动到最近点相距小于 1 厘米，持续至少 10 秒，最大速度为 2 厘米/秒。$\langle \text{DistanceGt} \rangle_{??}$； $(\langle \text{MoveSlowly} \rangle(A) \wedge \langle \text{DurationLong} \rangle)$； $\langle \text{距离小于} \rangle_{??}$
B	**口-口接触**：小鼠的鼻子相距小于 1.5 厘米。必须发生在接近后 5 秒内。$\langle \text{DistanceGt} \rangle_{??}$； $(\langle \text{缓慢移动} \rangle(A) \wedge \langle \text{持续时间长} \rangle)$； $\langle \text{DistanceLt} \rangle_{??}$； $\langle \text{DurationShort} \rangle$； $\left\langle \text{NoseNoseContact}(A,\ B) \right\rangle$
C	**口腔-生殖器接触**：小鼠的鼻子与尾根之间的距离小于 1.5 厘米。必须在接近后 5 秒内发生。$\langle \text{DistanceGt} \rangle_{??}$； $(\langle \text{缓慢移动} \rangle(A) \wedge \langle \text{持续时间长} \rangle)$； $\langle \text{DistanceLt} \rangle_{??}$； $\langle \text{DurationShort} \rangle$； $\left\langle \text{NoseGenitalContact}(A,\ B) \right\rangle$
D	**口腔-耳部接触**：小鼠的鼻子与耳朵之间的距离小于 1.5 厘米。必须在较短时间内发生在接近后超过 5 秒。$\langle \text{DistanceGt} \rangle_{??}$； $(\langle \text{缓慢移动} \rangle(A) \wedge \langle \text{持续时间长} \rangle)$； $\langle \text{距离小于} \rangle_{??}$； $\langle \text{持续时间短} \rangle$； $\left\langle \text{鼻-耳接触}(A,\ B) \right\rangle$
E	**追逐**：小鼠的移动速度超过每秒 15 厘米，最近点之间的距离小于 5 厘米，并且小鼠之间的角度偏差小于 30 度。$\left\langle \text{奔跑}(A) \right\rangle \wedge \left\langle \text{奔跑}(B) \right\rangle \wedge \langle \text{距离小于} \rangle_{??} \wedge \left\langle \text{朝向移动}(A,\ B) \right\rangle$
F	**注视**：小鼠之间的距离大于 5 厘米且小于 20 厘米，并且注视偏移为一只小鼠与另一只小鼠身体的夹角小于 15 度，最短持续时间为 3 秒。$\langle \text{距离小于} \rangle_{??} \wedge \langle \text{DistanceGt} \rangle_{??} \wedge \left\langle \text{凝视}(A,\ B) \right\rangle \wedge \langle \text{持续时间不短} \rangle$

C 神经网络基线方法

输入特征因数据集而异：对于单对象涩谷、单对象华沙和海上监视，状态由 x 位置、 y 位置和时间组成（状态维度为 3）。对于双对象华沙，状态由每个对象的 x 和 y位置、时间，以及一个用于指示每个对象是否存在于该状态中的掩码（1 或 0）组成。对于 MABe22，有两只小鼠，每只小鼠有 12 个经过跟踪的关键点，每个关键点都是一个 x、 y 对。

在主动学习期间，我们为预测为正类概率最高的未标记轨迹添加标签。

Transformer 我们训练一个 Transformer 模型，使用交叉熵损失直接预测类别标签。我们改编了来自 [29] 的轨迹预测模型，但将最后一层修改为预测类别逻辑值，而非未来位置。

长短期记忆网络 首先，我们在数据集的每条轨迹上训练一个仅解码器长短期记忆网络（改编自 Argoverse v1 [16] 的代码），以预测下一个状态。长短期记忆网络的嵌入维度是输入维度的两倍，隐藏状态维度是输入维度的四倍。我们训练它 1000 个训练轮次。用于逻辑回归的轨迹嵌入是该长短期记忆网络的最终隐藏状态。

D 证明

D.1 定理 2 的证明

由于我们按先进先出顺序弹出盒子，因此执行的是广度优先搜索。因此，随着 $N \to \infty$，搜索树中的深度也趋于无穷。所以，只需证明在每增加一层时，我们至少剪去空间中的一个常数比例。具体而言，每增加一层，我们至少剪去剩余空间的 $1/9$ ，且 $(1-1/9)^d \to 0$ 随着 $d \to \infty$。

对盒子 b 进行剪枝时，无论 b 的剪枝对是一致还是不一致，我们总会剪去 b_{lower}、 b_{upper} 和 b_{center}。这三个盒子位于 b 的中心线上，而且它们的中心线构成了 b 中心线的一个划分。因此，其中至少有一个（记作 b'）包含 b 中心线的 $\geq 1/3$。因此， b' 至少跨越 b 宽度的 $1/3$ 和高度的 $1/3$ ，面积至少为 $1/9$。由于在搜索树的每一层，每个盒子至少会被剪去 $1/9$，所以整体剩余空间也至少会被剪去 $1/9$。

D.2 定理 3 的证明

固定一个参数 $v \in \mathbb{R}^d$ 和一个正向量 $u \in \mathbb{R}^d_{>0}$。设 $t^*_{Q,z} := [\![Q]\!]^q_{v,u}(z)$ 以及 $f_{Q,z}(t) := [\![Q(t \cdot u + v)]\!](z)$。从概念上讲，$f_{Q,z}$ 考虑参数空间中从 v $(t=0)$ 到 $v+u$ $(t=1)$ 的线段，给出该线段上的点 t 是否位于 Q 和 z 的满足区域中。

¯我们称 t 是反单调函数 $f\colon\ \mathbb{R}\to\mathbb{B}$ 的边界值，如果对任意 $t'>t$， $f(t')=0$，并且要么 $t=-\infty$，要么 $f(t)=1$。注意，如果 t 是 $f_{Q,\ z}$ 的边界值，则 $t\cdot u+v$ 是 Q 和 z 的边界参数。因此，只需证明 $t^*_{Q,\ z}$ 是 $f_{Q,\ z}$ 的边界值。

首先，我们证明：如果 t_1^* 和 t_2^* 分别是 f_1 和 f_2的边界值，那么 $t^*=\min\{t_1^*,t_2^*\}$ 是 $f(t)=f_1(t)\wedge f_2(t)$的边界值——也就是说，合取的边界值是各边界值的最小值。假设 $t^*=t_1^*\le t_2^*$（另一种情况可用对称方法证明）。因此，对于任意 $t'>t^*, t'>t_1^*$, 有 $f_1(t')=0$, 从而 $f(t')=0$。此外，要么 $t_Q^*=-\infty$，此时结论成立；要么 $f_1(t^*)=f_1(t_1^*)=1$。后一种情况下，我们还有 $f_2(t_2^*)=1$，并且由于 f_2是反单调的， $f_2(t_Q^*)=1$。因此 $f(t^*)=f_1(t^*)\wedge f_2(t^*)=1$。类似的论证表明，$\max\{t_1^*,t_2^*\}$ 是 $f(t)=f_1(t)\vee f_2(t)$的边界值——也就是说，析取的边界值是各边界值的最大值。

现在我们通过对 Q 进行归纳法证明：$t^*_{Q,\ z}$ 是 $f_{Q,\ z}$ 对任意 z 的边界值：

对于 $Q=\varphi_{??i}$，根据定义，$t^*_{Q,\ z}=\frac{\iota\varphi}{(z)_{u_i}}-v_i$ 且 $f_{Q,\ z}(t)=\mathbb{1}(\iota\varphi(z)\ge t\cdot u_i+v_i)$。因此，~~$f_{Q,\ z}(t)=\mathbb{1}(\iota\varphi(z)_{u_i}-v_1\ge t)$~~，所以 $t^*_{Q,\ z}$ 是 $f_{Q,\ z}$ 的阈值。

对于 $Q=\varphi$，首先假设 $\underset{\varphi}{\text{sat}}\ z$ $_z=\infty$ z $=$ $()=1$，因此根据定义 t^*_Q，并且 f_Q，(t)。由于不存在 $t'>t^*_Q,\ z,\ t^*_Q,\ z$ 是 $f_Q,\ z$ 的边界值。现在假设 $\text{sat}\varphi(z)=0$，因此根据定义 $t^*_Q,\ z=-\infty$ 且 $f_Q,\ z(t)=0$。由于对于任意 $t'>t^*_Q,\ z,\ f_Q,\ z(t')=0,\ t^*_Q,\ z$ 是 $f_Q,\ z$的边界值。

对于 $Q=Q_1\wedge Q_2$, $t^*_{Q,\ z}=$最小值$\{t^*_{Q_1,z},t^*_{Q_2,z}\}$ 和 $f_{Q,\ z}(t)=f_{Q_1,\ z}(t)\wedge f_{Q_2,\ z}(t)$。根据归纳假设，$t^*_{Q_1,\ z}$ 是 $f_{Q_1,\ z}$ 的边界值，而 $t^*_{Q_2,\ z}$ 是 $f_{Q_2,\ z}$的边界值。因此，根据我们之前的结果，$f_{Q,\ z}$（合取）的边界值是 $t^*_{Q,\ z}$（最小值）。

对于 $Q=Q_1;\ Q_2$, $t^*_{Q,\ z}=$最大值$_{0\le k\le n}$最小值$\{t^*_{Q_1,z_{0:k}},t^*_{Q_2,z_{k:n}}\}$和 $f_{Q,\ z}(t)=\bigvee_{k=0}^n f_{Q_1,\ z_{0:k}}(t)\wedge f_{Q_2,\ z_{k:n}}(t)$。与之前一样，我们先应用归纳假设，然后应用关于合取和最小值的先前结果，最后应用关于析取和最大值的结果，从而得出 $f_{Q,\ z}$ 的边界值是 $t^*_{Q,\ z}$。