????????????

[illegible] unit [illegible]

- [illegible] [illegible] ——[illegible]
- [illegible] [illegible] ——[illegible] / [illegible] / [illegible] + [illegible] + [illegible]
- [illegible] [illegible] unit [illegible] [illegible] [illegible] "[illegible] → [illegible] " [illegible]

??????

[illegible] **ground truth [illegible] unit [illegible]** ——[illegible]

```
([illegible])[[illegible]]([illegible], [illegible], ...)
()[[illegible]]([illegible])
```

[illegible] `(M)[[illegible]](B)` [illegible] `(A)[[illegible]](B)` [illegible] ID[illegible] u_a / u_b / ...[illegible] §[illegible]

[illegible] ——`(X)` [illegible] `[P]` [illegible] `—` / `└─` [illegible] ——[illegible] unit [illegible] [illegible] **unit [illegible]** [illegible]

??????

[illegible] unit [illegible]

[illegible]	[illegible]
(X)	[illegible] node [illegible] (X) [illegible]
[P]	[illegible] unit [illegible] [P] [illegible]
—	[illegible] — [P] — [illegible] [P] — [illegible]
……	[illegible] `subject_mentions` [illegible]

[illegible] / [illegible] [illegible] §[illegible] [illegible] 4 [illegible] `S*` [illegible] ——[illegible] `(X)*` [illegible] `(X)` [illegible] unit [illegible]

??????

输入文本 → 输出 unit 集合 + 图拓扑。共五步，前三步沿用 认知图：以谓词作为概念 §五 抽取流程，后两步是本文档新增。

?? 1 — ??????????

参照原 §五 .1–2：识别小句 / 动词 / 论元 / 修饰 / 辖域；指代还原；特殊句式标准化（把字句、被字句、兼语、连动、是 / 有字句、比较句）；句类识别。

多句 / 多段输入 ：跨句指代和零指代在这一步统一还原。代词不能还原时降级为占位 node `?_1` / `?_2`，并打 `()[置信度](low)`。

?? 2 — ????

按因果 / 转折 / 时序 / 目的 / 条件 / 让步切分小句。保留连接关系，待嵌套层装回（原则 III）。

?? 3 — ?? unit ?

逐小句产出 unit：

- 选谓词（归 family 基础成员）；
- 定主体（ ≥1 个具体 node）；
- 补修饰行；
- 同时满足 §四 四原则 ——事实优先 / 最具体标识符 / 嵌套修饰 / 自洽。

派生 node `(M)[的](B)` 在使用前先声明，渲染名 `MB` 入图。同义命名 `(A)[同义](B)` 在两个标识符首次共现时声明。

派生 / 同义 不是默认操作 ——按以下规则触发：

- 派生：在两种情况下使用 ——（ a）现有标识符在当前图里有歧义、需要收窄（例： `复杂不完美的环境` ≠ 任意环境）；（ b）当前事实的重点就是 描述 / 分类一个实体 ——名词短语作主句论元（ "设想 X" / "讨论 X" / "考虑 X"）时， X 内部的修饰子句应当用派生节点层级表达，让 `[属于]` 把实例绑到类。
- 同义：两个 独立来源 的标识符首次共现（例： `论文-IAA ↔ 《深度强化学习的想象增强智能体》`、`想象增强智能体架构 ↔ 想象增强智能体` ）。不用同义给派生 node 起别名 ——那是 alias 不是 synonym，没有信息增益。

两种判别启发 ：

1. 重点是描述 vs 事件：去掉量词 "一位 / 一个 / 一种 "，看剩下的是不是个动词主导的完整句。是 → 事件型，主谓宾 + 修饰行；不是（剩下的是名词短语） → 描述型，派生层级 + `[属于]`。

2. 避免重复编码：写完一条派生 / 同义后，看是否同一事实在派生名、谓词、修饰行多处重复出现。是 → 简化掉冗余的层（要么删派生留事件，要么删事件留派生）。

反例 **A —** 描述错抽成事件：原文 "一位每天乘坐列车通勤的人"（名词短语）写成 `（通勤者）[通勤]() + [工具](列车) + [频率](每天)` ——把描述抽成动作，丢失 "她是什么人"的分类语义。正解：`（列车通勤）[的](人) + （每天）[的]（列车通勤的人） + （通勤者）[属于]（每天列车通勤的人）`。

反例 **B —** 派生 + 同义 + 事件四处重复：同句写成 `（列车通勤）[的](人) + （列车通勤的人）[同义]（列车通勤者） + （列车通勤者）[通勤]() + [工具](列车)` ——同义被滥用作 alias，"列车通勤" 编码四次。

段（抽取块）的边界：决定 `()` 指代的范围、决定局部 ID `u_a / u_b` 的有效域。一般以原文自然段或一两句一组划段；段终止后 `()` 不再有指代，必须重起一条主 unit；跨段引用必须用持久 unit ID。这套规则照搬 §二 .2–3，不在本文档重复。

?? 4 — ??????????????

遍历输入产出的所有 unit 集合 U（跨句、跨段统一收集，不分段重置）：

```
nodes  := ⋃ unit ∈ U {主体, 谓词, 论元, 所有修饰行的值}
S*     := ⋃ unit ∈ U {主体}                    # 含派生 / 同义 / 修饰行所属主 unit 的主体
preds  := { (unit, 谓词位置) }                 # 注意：以位置为单位，不去重
```

每个 node 的角色：

- `n ∈ S*` → 入口节点（`subject_mentions` 非空，可作为反向索引入口）
- 否则 → 仅论元节点（只在 `context_mentions` 中出现）
- 谓词位置一律渲染为 `[P]` ——同一谓词出现两次就是两个 `[P]` 实例（边的关系类型挂在边上，不共享节点）

派生 node 的基础 node `B`（出现在 `(M)[的](B)` 的论元位）：在本输入里若 B 没单独作过主体，仍然是仅论元节点；这条派生声明本身让 M 进 S*（M 是该 unit 的主体）。

?? 5 — ????

在本段产出的 unit 集中标记一条 **unit** 为核心 ——它是这一段的 narrative center。通常是：

- 命题态度 wrap（`[认为] / [认知] / [清楚] / [主张] / [怀疑]`）整段事实的那条；
- 或因果链顶层（最末端的 `[导致] / [造成]` unit）；
- 或一段就立人物 / 立场景的描述句，主 unit 是 `[属于] / [是]` 的分类断言。

其余 unit 自动是分支 ——它们仍然是独立 unit、有完整结构 / 反向索引 / 查询语义，只是叙事地位上从属于主 **unit**。

主线选定没有结构后果 ——它不改变图遍历语义，也不影响 unit 文本的 ground truth。它的作用是：

1. 视觉布局 ：核心 unit 横向居中排在图的主轴上，分支从主轴节点向下 /向上引出；
2. 段查询 ：问 "这段在讲什么 "直接读核心 unit，不必遍历全部 unit 综合；
3. 叙事中心化 ：让一段的核心断言和支撑细节在层级上有明确区分。

主线视觉边标签 **vs unit** 文本谓词 ：视觉图上可用自然连接词（ 平时 / 清楚 / 造成），便于人读； unit 文本里始终写 **family** 基础成员 （属于 / 认为 / 导致），便于查询。映射例：

视觉自然词	**unit** 文本 **family** 谓词
平时 / 通常 / 是	属于 / 是
清楚 / 知道 / 认为	认为 / 认知 / 知道
造成 / 引发 / 让	导致 / 造成
提出 / 介绍	(元话语，原则 降级，不上主线)

空段允许无主 **unit**：纯派生 / 同义声明段（如开头铺设词典）可能没有自然中心；这种段标记主 unit 为 `null` 即可。

?? 6 — ?????

每条 unit `(S)[P](A_1, ..., A_n)` 装配为一个星形子图：

```
(S) — [P] — (A_1)
          — (A_2)
          ...
          — (A_n)
```

修饰行 `()[mod-P](v)` 挂在主 unit 的 `[P]` 上：

```
(S) — [P] — (A)
          └— [mod-P] — (v)
```

嵌套 —— 当 A_i 是局部 ID（u_a / u_b / ...）：把内层 unit 的 `[P]` 直接接到外层 `[P]` 的论元位，跳过中间节点。读图者沿外层 `[P]` 走到内层 `[P]`，再展开内层 unit。

```
外层: (S)[目的是](u_a, u_b)
渲染: (S) — [目的是] — [u_a 的谓词] — ...
                     └— [u_b 的谓词] — ...
```

派生簇 —— `(M)[的](B)` 是一条普通 unit：

```
(M) — [的] — (B)
```

派生 node 的渲染名 `MB` 在后续 unit 里被引用时，复用 `(M)` 的位置 ——不复制 M 节点，整张图里 M 始终一个实例。

同义簇 —— `(A)[同义](B)` 同样作为 unit 渲染。查询时 ≡ 索引由 `同义`谓词派生（ §六 .2）；图上是显式的 `[同义]` 节点和两端的 `(X)`。

跨段连接 —— 当输入跨多个段时，节点（依靠全局唯一性， § 四）天然把不同段产出的星形子图缝合在一起：第一段把 X 作主体 `(X)`，第二段又用 X 作论元 `(X)`，那 `(X)` 同时出现在两段对应的子图里 ——它们共用同一个实例。

外发尾巴 —— 给每个入口节点画一条 `------` 尾巴，端点空悬，表示该 `(X)` 在全图层面有 `subject_mentions` 项可达本输入之外。如果该入口节点的所有 unit 都在本输入内被产出，尾巴只是占位；图被并入更大图后尾巴连到外部 unit。

?? / ???? — ??????

输入超过一段时，直接句对句抽取段边界容易和原文物理段对不齐、跨段桥接也靠不住。 先摘要再分段 更稳：

1. 摘要剥叙述外壳 ——评价词、频率词、"我们提出 / 我们展示 " 类元话语丢掉，留事实骨架。
2. 摘要每个分句 = 一段抽取块 ——分句天然是该段的 spine 候选，段边界对齐 narrative center。
3. 段间桥接由谓词承接 ——摘要叙述 "问题 → 替代 → 已有解 → 本工作 "的因果链时，`[目的是]` / `[属于]` / `[导致]` 自然落到段间。
4. 跨段引用用持久 ID——格式 `sN.uM`（段 N 内的 u_M 升格）；段终止后 `()` 不再指代。

摘要丢什么要心里有数 ：强度（ "通常 "）、程度（ "远远 "）、分立能力声明（ "合理样本 "）会被压平。下游若依赖这些细节，回原文补 `()[量化]` / `()[程度]` 修饰行或独立 unit。

完整工作示例见 §六 例 6 — 一篇 9 行论文摘要 → 4 段 11 主 unit + 11 派生 / 同义 + 4 修饰行；同输入按句对句抽取会膨胀到 7 段 16 主 unit。

?????

构建出的图必须满足：

1. 节点全局唯一性 ：同一 node 在整段输入产出的图里全局唯一——一个 `(X)` 实例。一个 node 在多句多段多条 unit 里出现，对应同一个实例。
2. 谓词独立性 ：同一谓词在不同 unit 里是不同 `[P]` 实例。 "两次说 '具有 '"对应两个 `[具有]` 实例。
3. 角色单调性 ：节点一旦进入 S*（成为入口节点），永远是入口节点，不会因后续 unit 倒退为仅论元节点。
4. 修饰行从属性 ：修饰行的 `[P]` 必须挂在主 unit 的 `[P]` 上，不能直接连主体 `(X)`—— 否则就该升格为独立主 unit。
5. 嵌套不绕路 ：内层 unit 在外层 unit 的论元位上以 "另一个 `[P]`"形态出现，禁止用占位 `(X)` 中转。

6. 段内局部 **ID** 不外泄 ： `u_a / u_b` 等局部 ID
   只在所属段内的论元位上出现；跨段引用必须替换成持久 unit ID 或具体节点名。
7. 每段恰有一条主 **unit**（或显式标 `null`）；其余 unit
   全是分支。主线没有结构后果，仅影响视觉布局和段查询入口。
8. 可视图严格 **(**主体 **)—[**谓词 **]—▶(**论元 **)**：图示画出来的每条边都必须对应一条主 unit
   ，箭头两端都是显式 `(节点)`。出现 `[谓词] X` 这种浮空标签 → 缺一个显式节点或一条派生 `(M)[的](B)`
   声明，必须补全。

违反任一条 → 抽取或装配出错。

????????

实际输出有两层。 主输出 **— unit** 文本 是 ground truth；分段、主 unit 用注释标记：

```
# 段 1   spine: u1
(想象增强)[的](智能体)
(想象增强智能体)[需要提升](学习效率)          # u1
()[范围](复杂不完美环境)
()[工具](想象力)
()[证据](DeepMind)

# 段 2   spine: u3
...
```

派生输出 **—** 拓扑摘要 （可选，给可视化渲染器消费，由主输出自动派生、不持久化）：

```
graph:
  nodes:
    - id: 想象增强智能体
      entry: true                    # 入口节点（S* 成员）
      out_count: 4
    - id: 学习效率
      entry: false                   # 仅论元节点
    ...
  preds:
    - id: u1
      subject: 想象增强智能体
      predicate: 需要提升
      args: [学习效率]
      modifiers:
```

```
        范围： 复杂不完美环境
        工具： 想象力
        证据： DeepMind
    ...
  segments:
    - id: seg-1
      spine: u1                          # 段的主 unit；null 表示无主线
      branches: [u2, u3, u4, ...]        # 该段其余 unit
    ...
```

两者必须一致 ——若不一致，以 unit 文本为准。

????

? 1 — ??????

原文： 深蓝击败卡斯帕罗夫。

unit 文本：

```
（深蓝）[击败]（卡斯帕罗夫）
```

节点角色： S* = {深蓝 } → 入口节点；卡斯帕罗夫 → 仅论元节点； `[击败]` 是谓词实例。

拓扑示意：

```
            ---------- （外发）
              \
            （深蓝） —— [击败] —— （卡斯帕罗夫）
```

? 2 — ???? + ???

原文： **DeepMind** 通过赋予智能体想象力，提升了其在复杂不完美环境中的学习效率。

unit 文本（取自 svo_demo_0_processed.txt 句 1）：

```
（复杂不完美）[的]（环境）
（想象增强）[的]（智能体）
```

```
(想象增强智能体)[具有](想象力)
()[证据](DeepMind)


(想象增强智能体)[需要提升](学习效率)
()[范围](复杂不完美环境)
()[工具](想象力)
()[证据](DeepMind)
```

节点角色: S* = {复杂不完美 , 想象增强 , 想象增强智能体 }

node	角色	理由
复杂不完美	入口	派生声明 `(复杂不完美)[的](环境)` 的主体
想象增强	入口	派生声明 `(想象增强)[的](智能体)` 的主体
想象增强智能体	入口	多次作主体（具有 / 需要提升）
环境	仅论元	仅作论元
智能体	仅论元	仅作论元
想象力	仅论元	论元 + 修饰行值
学习效率	仅论元	仅作论元
复杂不完美环境	仅论元	修饰行值（派生 node 渲染名）
DeepMind	仅论元	修饰行值（证据）

拓扑示意:

```
(复杂不完美) ── [的] ── (环境)


(想象增强) ── [的] ── (智能体)


                                 -------- (外发)
                                 \
   (想象增强智能体) ── [具有] ── (想象力)
                     |          └── [证据] ── (DeepMind)
                     |
                     └── [需要提升] ── (学习效率)
                                   ├── [范围] ── (复杂不完美环境)
                                   ├── [工具] ── (想象力)
                                   └── [证据] ── (DeepMind)
```

匹配手绘图的形态：入口节点（带 `------` 尾巴）是反向索引登记点，手臂上是 `[P] — (X)`，多条 unit 通过共享 `(X)` 相连。

? 3 — ?????????

原文： **DeepMind** 提出 **X**，旨在应对真实世界的复杂性，从而提升深度强化学习的性能。

unit 文本（原则 I 元话语降级 + 原则 III 意图链嵌套）：

```
(真实世界)[具有](复杂性)                    # u_a
(深度强化学习)[需要提升](性能)              # u_b

(X)[目的是](u_a, u_b)
()[证据](DeepMind)
```

节点角色： S* = {真实世界 , 深度强化学习 , X} → 入口节点；其余 → 仅论元节点。

拓扑示意（嵌套用 `[P]` 直连 `[P]`）：

```
(真实世界) — [具有] — (复杂性)
                          ↑
                          | (作为论元位)
(X) — [目的是] ———————————┤
        |                 ↓
        |      (深度强化学习) — [需要提升] — (性能)
        └— [证据] — (DeepMind)
```

注意 `[目的是]` 的两个论元位接的是 另外两个 **`[P]`**（ u_a / u_b 的谓词），不是中间套 `(X)`——这是嵌套的几何特征。

? 4 — ?????????

原文（两句一段）： **DeepMind** 提出了想象增强智能体。该智能体在复杂环境中具有更高的学习效率。

unit 文本（两段抽取块，每段独立局部 ID 域；节点跨段共享）：

```
(想象增强)[的](智能体)
(想象增强智能体)[同义](DeepMind 提出的想象增强智能体)
()[证据](DeepMind)
```

```
(复杂)[的](环境)


(想象增强智能体)[具有](学习效率)
()[范围](复杂环境)
()[程度](更高)
```

节点角色（全局收集）：

- 入口节点：想象增强、想象增强智能体、复杂
- 仅论元节点：智能体、 DeepMind、环境、学习效率、复杂环境

`想象增强智能体`在第一段是主体（同义声明）、第二段还是主体（具有） ——它是同一个 `(想象增强智能体)`，把两段的子图缝合到一起。这是 §四 不变量 1 在跨句场景的直接体现。

? 5 — ??????????? + ????

原文：

> 设想一位每天乘坐列车通勤的人。大多数早晨，她的列车都能准时发车，她能轻松有备地赶上早会。但她清楚，一旦有一点意外发生 ——机械故障、信号失灵，甚至只是碰到雨天 ——都可能扰乱她原本的模式，造成迟到或慌乱。

这一整段的核心 narrative arc 是她清楚一旦意外发生会扰乱模式造成迟到 / 慌乱 ——句 1 描述她是什么人是 setup，句 2 立常态也是 setup，句 3 是核心断言。整段抽成 一条主线 + 多条分支，主线由命题态度 wrap 嵌套因果链。

???spine?

整段共 3 条 spine units，对应视觉主轴上的 3 条边：

```
# SPINE
(她)[属于](每天列车通勤的人)                # spine.1    视觉边 [平时]
(她)[认为](u_c)                                  # spine.2    视觉边 [清楚]
(u_b)[导致](迟到, 慌乱)                      # spine.3 = u_c    视觉边 [造成]
()[模态](可能)
```

视觉边标签 `平时` / `清楚` / `造成`来自原文， unit 文本里规范化到 family 谓词 `属于` / `认为` / `导致`。

??????spine.2 / spine.3 ???

```
(意外)[发生]()                                  # u_a
()[量化](一点)

(u_a)[扰乱](通勤者原本模式)                    # u_b
()[模态](可能)

(u_b)[导致](迟到, 慌乱)                        # u_c   ← 同 spine.3
()[模态](可能)
```

`(她)[认为](u_c)` 通过 u_c → u_b → u_a 的反向链，自动覆盖整条因果链——不需要写成 `[认为](u_a, u_b, u_c)` 把每节点都列出来。

????? unit???????????

```
# 分支：(她) 的描述（D 决定）
(列车通勤)[的](人)
(每天)[的](列车通勤的人)

# 分支：意外的子类（C 决定）
(机械故障)[属于](意外)
(信号失灵)[属于](意外)
(雨天)[属于](意外)

# 分支：(她) 的原本模式
(她)[的](原本模式)

# 分支：常态背景（句 2）
(列车)[发车]()
()[方式](准时)
()[量化](大多数早晨)
()[模态](能)

(她)[赶上](早会)
()[方式](轻松, 有备)
()[模态](能)
()[量化](大多数早晨)
```

????

按 "主轴 + 各节点分支 "整合排列，全部以 unit 文本表达。修饰行 `()[mod-P](v)` 缩进 2 格挂在上方最近的一条主 unit 的 `[P]` 上；视觉边标签写在 `#` 注释里，不进入语法。

```
# 主轴 spine
(她)[属于](每天列车通勤的人)                # spine.1   视觉: 平时
(她)[认为](u_c)                                  # spine.2   视觉: 清楚
(u_b)[导致](迟到, 慌乱)                      # spine.3 = u_c   视觉: 造成
  ()[模态](可能)

# spine.2 内部嵌套链
(意外)[发生]()                                  # u_a
  ()[量化](一点)
(u_a)[扰乱](通勤者原本模式)              # u_b
  ()[模态](可能)
(u_b)[导致](迟到, 慌乱)                      # u_c

# (她) 节点向下分支(D 决定)
(列车通勤)[的](人)
(每天)[的](列车通勤的人)
(她)[的](原本模式)

# (她) 节点的常态分支
(她)[赶上](早会)
  ()[方式](轻松, 有备)
  ()[模态](能)
  ()[量化](大多数早晨)

# (列车) 节点的常态分支
(列车)[发车]()
  ()[方式](准时)
  ()[量化](大多数早晨)
  ()[模态](能)

# (意外) 节点向下分支(C 决定)
(机械故障)[属于](意外)
(信号失灵)[属于](意外)
(雨天)[属于](意外)
```

??????

1. 主线选定 **— A.a** 单一核心断言 ：本段是说服性叙述，核心是 "她清楚一条规则 "——主 unit 选 `(她)[认为](u_c)`。属于 / `导致`也上主线是因为它们共享 `(她)` 起点和 `(迟到, 慌乱)` 终点，构成完整 narrative arc; 不是另立两段。

2. **B** 主线规范化 ：视觉用 `平时 / 清楚 / 造成` 让人读得自然； unit 文本写 family 谓词 `属于 / 认为 / 导致` ，让查询沿 family 闭包仍能命中。
3. **C** 子类独立 ： `机械故障 / 信号失灵 / 雨天` 写成 3 条 `[属于]` unit（不压成 （意外 ）节点的 `[子类](机械故障, 信号失灵, 雨天)` 修饰行），保证 "意外的所有子类 " 是图遍历一步。
4. **D（她 ）**的描述独立 ：派生 + `[属于]` 链立人物 ——不降为 （她 ）节点的 `[每天行为](乘列车)` 修饰行。这条选择保证 "每天列车通勤的人 " 作为类，能挂上 （她 ）赶上__ 等其他实例的常态属性供继承。
5. `(她)[认为](u_c)` 单论元嵌套 ： u_c → u_b → u_a 反向可达，主线只需写终点 unit ID；不重复 `[认为](u_a, u_b, u_c)` 。
6. 常态背景属于分支 ：句 2 的 `(列车)[发车]()` / `(她)[赶上](早会)` 不上主线 ——它们是 setup ，不是核心断言。视觉上挂在 （她 ）/（列车 ）节点旁。
7. 元话语 `设想`：原则 I 降级，不写 `(读者)[设想](...)` 。整段假说性可挂 `()[置信度](假设)` 给主 unit。
8. 跨句指代 `她`：步骤 1 还原为 （她 ）（直接作为主体 ID）；图里没有 "她 " 这个代词节点。

??????

入口节点	来源
列车通勤	派生主体 `(列车通勤)[的](人)`
每天	派生主体 `(每天)[的](列车通勤的人)`
她	中心节点 ——属于 / 认为 / 赶上 / 派生（原本模式）
列车	发车
机械故障 / 信号失灵 / 雨天	三条 `属于` unit 的主体
意外	发生

仅论元节点：人、列车通勤的人、每天列车通勤的人、早会、原本模式、通勤者原本模式（即 "她的原本模式 " 渲染名）、迟到、慌乱、修饰行值（每天、准时、轻松、有备、大多数早晨、一点）。

? 6 — ?????4 ??????? + ????

原文（ GAN-based domain adaptation 论文摘要）：

> 使用带有精心注释的图像数据集来训练现代机器学习算法，对于许多任务来说都是非常昂贵的。一个很有吸引力的替代方案是渲染合成数据，其中 ground-truth 注释会自动生成。不幸的是，纯粹在渲染图像上进行训练的模型通常不能推广到真实图像。为了解决这个缺点，此前有研究引入无监督的领域自适应算法，尝试在两个域之间映射表示，或者学习提取不变的特征。在这项工作中，我们提出了一种新的方法，以无监督的方式学习实现从一个域到另一个域在像素空间上的转换。我们基于生成对抗网络（ GAN ）的模型能够适应源域映像，就像从目标域中绘制的一样。我们的方法不仅产生了合理的样本，而且在某些无监督的域适应场景中也远远胜过最先进的技术。最后，我们展示了适应过程生成了在训练过程中没有见过的物体类别。

?????????

[illegible] "[illegible] / [illegible] / [illegible] / [illegible] "[illegible]

1. [illegible] [illegible] ML [illegible]
2. [illegible] [illegible]
3. [illegible] [illegible] ——[illegible]
4. [illegible] [illegible] GAN [illegible] → [illegible] UDA [illegible] SOTA[illegible]

[illegible] = 4 [illegible] spine[illegible]

????

[illegible] / [illegible] unit[illegible]

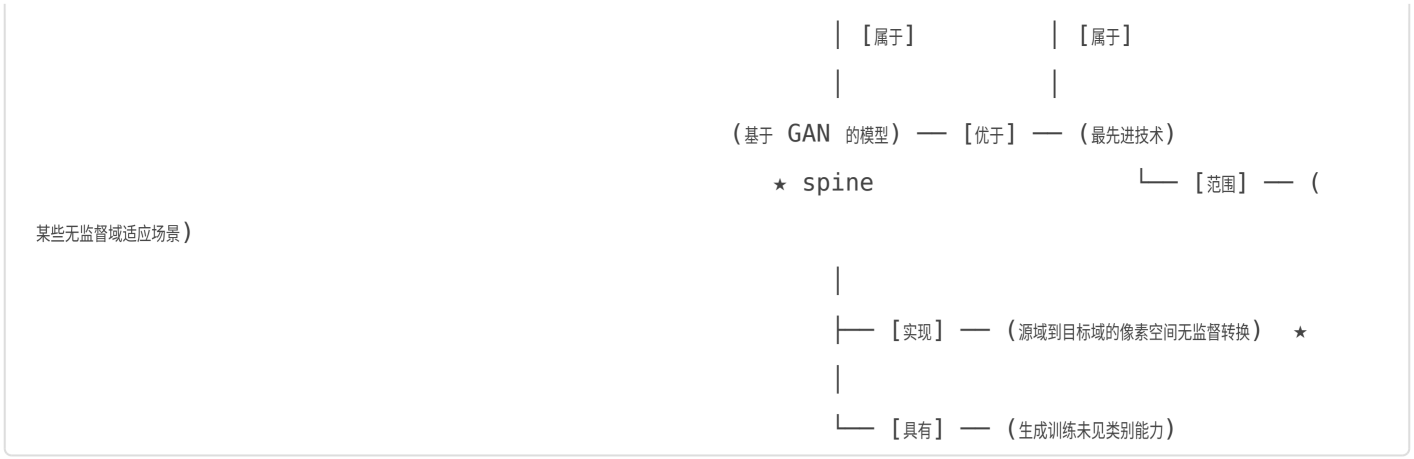
```
[illegible] / [illegible]
   ([illegible]) ── [[illegible]] ── ([illegible])            ([illegible]) ── [[illegible]] ── ([illegible])
   ([illegible]) ── [[illegible]] ── ([illegible])            ([illegible]) ── [[illegible]] ── ([illegible])
   ([illegible]) ── [[illegible]] ── ([illegible])            ([illegible]) ── [[illegible]] ── ([illegible])
   ([illegible]) ── [[illegible]] ── ([illegible])            ([illegible]) ── [[illegible]] ── ([illegible])
   ([illegible]) ── [[illegible]] ── ([illegible])            ([illegible] GAN) ── [[illegible]] ── ([illegible])
   (GAN) ── [[illegible]] ── ([illegible])



([illegible]) ── [[illegible]] ── ([illegible]) ── [[illegible]] ── ([illegible])
                                                 │
                                                 │ [[illegible]]
                                                 ▼
                                    ([illegible])
                                                 ▲
                                                 │ [[illegible]]
                                                 │
                                    ([illegible]) ── [[illegible]] ── ([illegible] ground-truth [illegible])



([illegible]) ── [[illegible]] ── ([illegible])  ◀── [illegible] u_e
                                                 ▲
                                                 │ [[illegible]]   ([illegible] u_e [illegible] [[illegible]][illegible])
                                                 │
      ([illegible], [illegible]) ◀── [[illegible]] ── ([illegible])
                                                         ▲              ▲
```



整篇 spine 落在 `(基于 GAN 的模型) ── [实现] ── (源域到目标域的像素空间无监督转换)`——本工作的 headline 断言。共 12 主 unit + 4 修饰行 + 11 派生 / 同义。

证据修饰行未画在图上，规则统一：段 3 (无监督领域自适应算法) 系列 `()[证据](此前研究)`；段 4 (基于 GAN 的模型) 系列 `()[证据](本工作)`。

????

5 条 unit 把 4 段缝起来：

- u2.1 `(渲染合成数据)[属于](精心注释图像数据集的替代方案)` — 段 2 ↔ 段 1，靠派生节点 `(精心注释图像数据集)[的](替代方案)` 链回。
- u3.1 `(无监督领域自适应算法)[目的是](s2.u2)` — 段 3 → 段 2，UDA 的目的就是解决段 2 的 u_e。
- u4.1 `(基于 GAN 的模型)[属于](无监督领域自适应算法)` — 段 4 → 段 3。
- u4.5 `(最先进技术)[属于](无监督领域自适应算法)` — 段 4 → 段 3。

沿 u4.1 / u4.5 的 `[属于]` 闭包，u3.1 的 `[目的是 u_e]` 自动继承到 (基于 GAN 的模型) 与 (最先进技术)——无须再画显式 `[目的是 u_e]` 边。

??????

1. 元话语降级：“我们提出 / 我们展示 / 此前有研究” 全按原则 I 降到 `()[证据]`（“本工作” / “此前研究”）。
2. 摘要分句 = 段 **spine**：4 段对应 4 条 spine；整篇 spine 落在 u4.2。
3. 跨段引用用 `sN.uM`：u3.1 引用段 2 的 u_e；u_a / u_b 局部 ID 不能跨段，跨段必须升格为持久 ID。
4. 派生节点搭概念坐标系：(精心注释图像数据集) / (渲染合成数据) / (基于 GAN 的模型) 都是派生节点，让跨段引用能落到具体标识符。

??????

原文表达	丢失的认知内容	严格抽取应补
“通常不能泛化”	强度（“通常” ≠ “一定”）	`()[量化](大多数)` 或 `()[置信度]`
“远远胜过”	比较程度	`()[程度](远远)`
“合理的样本”	一项独立的能力声明	单独 unit `(基于 GAN 的模型)[具有](合理样本生成能力)`

□□□

Updated 2026-05-09 15:15:52 UTC by Colin