NFL Big Data Bowl 2026 - Prediction解决方案报告

1) 比赛理解

1.1 背景与目标

长传是美式橄榄球里最具不确定性的瞬间：当球在空中时，可能发生达阵、拦截或争夺接球。NFL 希望更好地理解从**球被抛出**到**被接住或判定为不完整传球**这段时间内，进攻与防守球员如何移动。

在预测赛道中，NFL 提供了**出手前的追踪序列**，并在四分卫**释放球**时刻截断；同时给出**目标接球者（target）**与**传球落点（ball landing location）**。参赛者需要建立模型，预测球在空中时每名球员的后续轨迹，输出越接近真实轨迹的方案排名越高。

1.2 竞赛数据

> 训练数据按周分文件，包含**传球前输入序列**与**传球后输出轨迹**；测试集提供传球前输入与需要预测的行模板。

文件结构

- `train/`
  - `input_2023_w[01-18].csv`：传球前追踪数据（特征输入）
  - `output_2023_w[01-18].csv`：传球后追踪数据（监督标签）
- `test_input.csv`：保留测试比赛的传球前追踪数据（无真值）
- `test.csv`：需要预测的 ((game_id, play_id, nfl_id, frame_id)) 行模板
- `sample_submission.csv`：提交格式示例（`id,x,y`）

核心字段（input）

- 标识：`game_id`, `play_id`, `nfl_id`, `frame_id`
- 场地/回合：`play_direction`, `absolute_yardline_number`
- 球员属性：`player_height`, `player_weight`, `player_birth_date`, `player_position`
- 阵营与角色：`player_side`（Offense/Defense），`player_role`（Defensive Coverage / Targeted Receiver / Passer / Other Route Runner 等）
- 追踪状态：`x`, `y`, `s`（速度），`a`（加速度），`o`（身体朝向），`dir`（运动方向）
- 预测长度：`num_frames_output`（该球员需要预测多少帧）
- 传球落点：`ball_land_x`, `ball_land_y`

输出字段（output / 训练标签）

- 标识：`game_id`, `play_id`, `nfl_id`, `frame_id`
- 目标：`x`, `y`（传球后每一帧的真实位置）

关键数据约定

- 追踪数据频率为 **10Hz（每秒 10 帧）**，例如球在空中 2.5 秒 ≈ 25 帧需要预测。

1.3 评估指标（RMSE）

比赛以预测轨迹与真实轨迹之间的**均方根误差**（RMSE）评估（同时考虑 (x) 与 (y) 两个方向）：

$$RMSE = \sqrt{\frac{1}{2N}\sum_{i=1}^{N}\left((x_{true,i}-x_{pred,i})^2+(y_{true,i}-y_{pred,i})^2\right)}$$

直观理解

- 你的模型输出每个需要预测的点 ((x,y))；
- RMSE 越小越好，等价于“平均每个点的二维位置误差越小”。

1.4 领域知识入门（面向不懂橄榄球的同学）

- **进攻方**：目标接球者（Targeted Receiver）通常要跑到**落点附近**完成接球；其他跑路线者会拉开空间、吸引防守。
- **防守方**：防守覆盖者（Defensive Coverage）经常不是“追落点”，而是“盯人/跟路线”，尽量在落点附近干扰接球或争抢。
- **四分卫出手前最后几帧**的信息最关键：速度、方向、人与人相对位置，往往决定出手后 1–3 秒的运动趋势。

坐标系怎么理解

- (x)：沿球场长度方向，范围约 ([0,120]) 码（包含两端端区）。
- (y)：沿球场宽度方向，范围约 ([0,53.3]) 码。
- 所以推理阶段会把预测裁剪到这两个范围内（防止“跑出场外”的非法点）。

小型术语词汇表（10 条）

1. **Play（回合/一次进攻）**：从开球到哨响的一段动作序列。
2. **Pass play（传球回合）**：四分卫选择传球的回合。
3. **Ball in the air（球在空中）**：从出手到接到/未接到的时间段，本赛题预测区间。
4. **Targeted Receiver（目标接球者）**：四分卫主要传向的那名接球目标。
5. **Defensive Coverage（防守覆盖）**：主要负责盯防接球路线/接球人的防守球员。
6. **Route（跑路线）**：接球手预先设计的移动路线（直线、斜切、横移、急转等）。
7. **Tracking data（追踪数据）**：每帧记录球员位置、速度、方向等（本题 10Hz）。
8. **(s) / (a)**：速度（码/秒）与加速度（码/秒(^2)）。
9. **(o) vs (dir)**：(o) 更像身体朝向，(dir) 更像运动方向（代码里用 (dir) 分解速度）。
10. **RMSE**：二维坐标误差的综合指标，越小越好。

---

2) 解决方案解析

2.1 方案流程概览

A. 数据预处理 / 特征工程（`data_fe.py`）

- 将 (s,dir) 分解为 ((v_x,v_y))，并构造动量、动能、朝向差等"物理启发"特征。
- 构造**对手交互**（最近对手距离、逼近速度、3/5 码内对手数）。
- 为防守覆盖者构造**镜像接球者（mirror）**：近似"我在盯谁"。
- 用末尾 5 帧做**路线形状聚类（KMeans, K=7）**，得到离散 `route_pattern`。
- 在出手前最后一帧做**邻域聚合（GNN-lite）**：半径内近邻的相对位移/相对速度差做加权汇总。
- 构造大量**时序派生**（lag/rolling/std/delta/EMA）。
- 最重要：构造**几何终点基线（geo endpoint）**及其派生几何特征（让模型学"偏差修正"而非从零学轨迹）。

B. 模型设计与训练（`models.py`, `train.py`）

- 输入：传球前最近 (T=10) 帧，特征维度约 (F\approx 150+)。
- 模型：**GRU(2层, hidden=128) + 注意力池化 + MLP head**，输出未来 (H=94) 步的二维位移序列。
- 训练目标：预测相对位移 ((dx,dy))，再通过 `cumsum` 形成轨迹。
- 损失：**Temporal Huber**（Huber + 时间衰减权重 + mask 处理变长未来长度）。
- 验证：**GroupKFold 按 game_id 分组**避免同一场比赛泄漏到验证集。

C. 推理与后处理（`inference.py`）

- 对测试序列用每折 scaler 标准化后推理，**5 折模型均值集成**。
- 将预测的相对位移加回"输入窗口最后一帧的 ((x,y))"得到绝对坐标。
- 输出裁剪到场地范围：(x\in[0,120], y\in[0,53.3])。

2.2 关键技术点

2.2.1 模型选择与原因

- **GRU**适合短窗口（10 帧≈1 秒）动力学建模：能抓住"速度/方向/加速度"的连续变化。
- **注意力池化**相当于可学习的"挑重点帧"：很多情况下出手前最后 2–3 帧最关键，它能比简单取最后隐状态更稳。
- **输出增量 + 累加（cumsum）**把复杂轨迹拆成小步预测：减少直接预测绝对位置的难度，训练更稳定。

2.2.2 特征工程

1. **物理/运动学特征**

- ((v_x,v_y))、动量、动能、朝向差（(o-dir)）等，让模型更容易学"惯性 + 转向"。

2. **落点相关几何特征（ball geometry）**

- 到落点距离、指向落点单位向量、沿落点方向的 closing speed（是否在接近落点）。
- 直觉：目标接球者往往"向落点收敛"，防守者也常在落点附近收缩。

3. **对抗交互特征（opponent / pressure / mirror）**

- 最近对手距离 + 逼近速度：刻画“压力/被贴身程度”。
- **mirror 接球者**：对 Defensive Coverage 来说，“盯人”比“追球”更常见；镜像能把“我跟谁绑定”显式化。

4. **路线形状聚类（route_pattern）**

- 用末尾 5 帧提取“直线性、转向幅度、纵深/横向位移、速度变化”后聚类。
- 给模型一个离散语义：这名球员出手前更像“直冲/横移/急转”，未来分布自然不同。

5. **邻域聚合（GNN-lite embedding）**

- 只在最后一帧构图：用半径内近邻的相对位移/速度差加权汇总，并区分友军/对手通道。
- 本质是把“周围局势”压缩成定长向量，避免“场上 22 人输入维度爆炸 + 顺序不固定”。

2.2.3 训练策略

- **变长未来长度**：统一 pad 到 (H=94)，并用 mask 只在真实步长上算 loss。
- **Temporal Huber + 时间衰减**
  - Huber 抗异常点：小误差近似 MSE、大误差近似 MAE。
  - 时间衰减 $w_t = \exp(-\lambda t)$ 让模型更关注“更可预测、更重要”的近未来轨迹段。
- **GroupKFold（按 game_id）**
- 同一场比赛的战术风格/对阵队伍/场地因素会相似，分组能更接近真实泛化。
- **每折独立 StandardScaler**
  - 仅用训练折拟合标准化，避免验证信息泄漏。

2.2.4 这份方案的“创新点”与机制解释

创新点：几何/规则终点（geo endpoint）+ 让模型学残差的思路

- 代码先用一套可解释规则给出“合理终点”：
  - 普通球员：按当前速度匀速外推到未来总时长；
  - 目标接球者：终点直接设为落点；
  - 覆盖者：若找到 mirror 接球者，则终点≈落点 +（覆盖者相对 mirror 的偏移）。
- 再派生出一组 `geo_*` 特征：指向终点的向量、距离、所需速度/加速度、对齐度等。

为什么这会提升 RMSE（机制层面）

- 轨迹预测最难的是“绝对位置分布很宽”：同样的出手前状态，未来可能分叉。
- 几何终点把“战术语义”强行注入：
  - 接球者**必然**朝落点附近收敛；
  - 覆盖者与接球者存在“绑定关系”；
  - 其他人至少符合“惯性外推”的物理先验。

2.3 代码解析（按模块）

2.3.1 `data_fe.py`

- `get_velocity(s, dir)`：把速度标量按方向角分解为 ((v_x,v_y))。
- `get_opponent_features(input_df)`：
  - **只取末帧**计算对抗关系，输出每个球员一行聚合特征：最近对手距离、逼近速度、近邻对手计数；以及覆盖者的 mirror 接球者速度与相对偏移。
- `extract_route_patterns(input_df, fit=...)`：
  - 末尾 5 帧 -> 轨迹形状统计 -> `StandardScaler + KMeans(n_clusters=7)` -> `route_pattern`。
- `compute_neighbor_embeddings(..., k_neigh=6, radius=30, tau=8)`：
  - 末帧构图，半径内取最近邻；权重$w = \exp(-dist/\tau)$；友军/对手双通道聚合；输出固定维度邻域向量 + 距离统计。
- `compute_geometric_endpoint / add_geometric_features`：
- 生成 `geo_endpoint_x/y` 及其派生几何特征（向量、距离、对齐度、所需速度/加速度等）。
- `prepare_sequences_geometric(...)`（管线核心）：
  - 汇总所有特征 → 选取最后 `window_size=10` 帧作为输入序列 ([T,F])
  - 训练目标：未来每帧相对位移序列（相对"输入窗口最后一帧"）
  - 返回：`sequences, targets_dx, targets_dy, sequence_ids`，以及推理复用的 `route_kmeans/scaler`。

2.3.2 `models.py`

- `JointSeqModel(input_dim, horizon)`：
  - GRU 输出整段隐状态 ([B,T,128])
  - 可学习 query 做注意力池化得到全局向量 ([B,128])
  - MLP 输出 ([B,H,2]) 的"每步增量"，最后 `cumsum` 得到轨迹。
- `TemporalHuber(delta=0.5, time_decay=0.03)`：
  - Huber + 时间权重 + mask（pad 的部分不计入 loss）。

2.3.3 `train.py`

- `prepare_targets(...)`：把每个样本不定长 ((dx,dy)) pad 到 (H=94)，并构造 mask。
- `train_model(...)`：
  - AdamW + 梯度裁剪 + ReduceLROnPlateau + EarlyStopping
  - 训练/验证 batch 预先打包成 tensor，减少每个 epoch CPU 开销。
- 主流程：
  - 读入 `w01~w18` 合并训练集
  - 构造序列
  - `GroupKFold(n_splits=5, groups=game_id)` 训练 5 个模型与 5 个 scaler
  - 落盘：`exp_models_scalers.pkl`、`route_kmeans_scaler.pkl`

2.3.4 `inference.py`

- 读取 `test_input.csv` + `test.csv`（模板决定需要预测哪些帧）
- 构造测试序列（复用训练得到的 route 聚类器）
- 5 折模型分别推理并取均值集成
- 绝对坐标 = 输入末帧 ((x{*last*},*y*{last})) + 预测相对位移
- 裁剪到合法场地范围并写出 `submission.csv`

---

3) 简历项目模板

Kaggle — NFL Big Data Bowl 2026: Prediction（2025.11–2026.01） | LB RMSE 0.xxx

- **背景/挑战**：基于 NFL Next Gen Stats 10Hz 追踪数据，给定出手前短序列与传球落点，预测球在空中期间多名球员未来 ((x,y)) 轨迹（变长输出、强交互、强战术先验）。
- **解决方案：**
  - **战术先验注入**：构建"几何终点基线（geo endpoint）"，对目标接球者强制收敛落点、对覆盖者引入盯人镜像偏移，用可解释规则缩小未来轨迹分布并提升可学习性。
  - **交互结构建模**：在出手前末帧构建对抗/邻域特征（最近对手距离、逼近速度、半径近邻加权聚合），将 22 人局势压缩为定长向量，增强对"压力/掩护/拥挤度"的感知。
  - **序列建模稳定性**：使用 GRU 编码 + 注意力池化输出全局状态，预测增量并累加成轨迹；采用 Temporal Huber + 时间衰减与 mask 处理变长预测区间。
- **量化结果**：5-fold GroupKFold（按 game_id 防泄漏）+ 折均值集成，最终 **RMSE 0.xxx**。
- **技术栈**：Python, PyTorch, scikit-learn（StandardScaler/KMeans/GroupKFold）, 时序特征工程（lag/rolling/EMA）, 轨迹预测（sequence-to-sequence, delta+cumsum）。