NNBranchPredictor

基本结构

- Path:用于存放历史; path_length:128

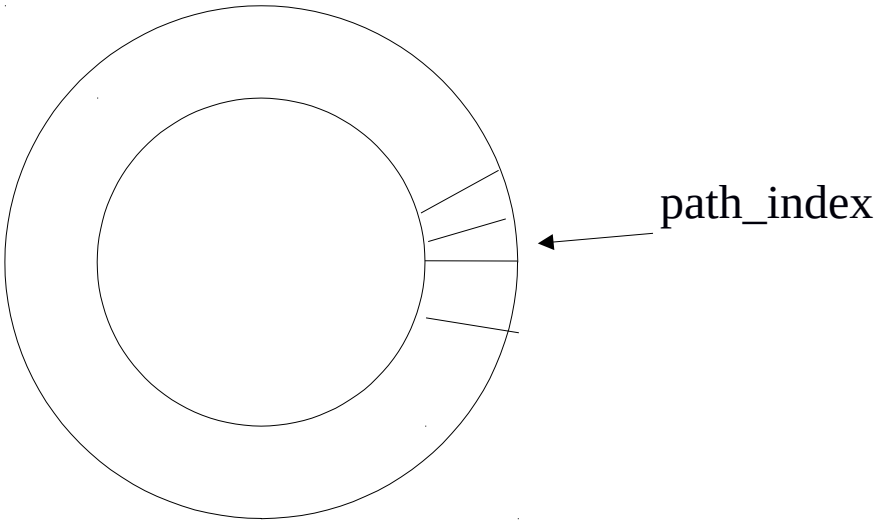
128 个历史

Path[0]	Path[1]	... ...	Path[126]	Path[127]

128 个 PathInfo

```
struct PathInfo{
    bool taken; //是否跳转
    IntPtr target; // 64 历史目标地址
};
```

- 左侧新历史进，右侧旧历史出。
- 实现方式采用循环队列，由 path_index 指向当前位置



- 优势是防止内存拷贝

- bias_weights: 偏向权重表; num_bias_entries: 2048 行

2048 行
取值范围[-128,127]

[-128,127]
[-128,127]
... …
[-128,127]

- weights:权重表; num_entries: 512 行

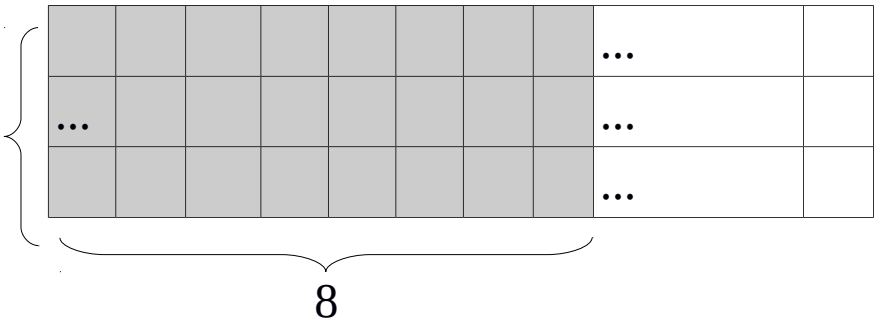
128 列，对应 128 个历史的权重

512 行

[-128,127]	... ...	[-128,127]
[-128,127]	... ...	[-128,127]
... ...	... ...	... ...
[-128,127]	... ...	[-128,127]
[-128,127]	... ...	[-128,127]

- 每 8 列为一个小表

512



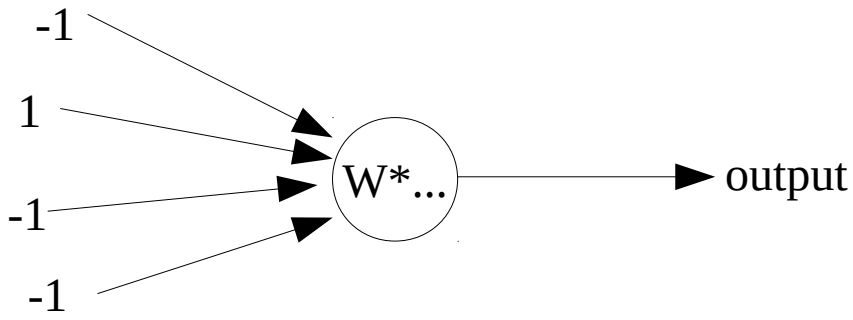
- theta:阈值，用来判定训练是否成熟
- tc:用于自适应更新阈值的计数器
- bias_coefficient:偏向系数，对应偏向权重
- coefficients:偏向系数，对应 128 个历史权重

128 个系数

coefficients[0]	coefficients[1]	... ...	coefficients[127]

- 权重距离当前越近，系数越高

预测方法



输入参数:ip

步骤:

- 索引偏向权重行号（bias_hash）

  bias_index = (ip >> 4) % num_bias_entries;

- 找出权重表行号

  每 8 列索引一次（index_entries），每次索引的小表行号是不一样的:

  indexes[0] =

  (ip >> 4) ^ ((path[0].target << 4) ^ … ^ (path[7].target << 4));

  …

  indexes[15] =

  (ip >> 4) ^ ((path[120].target << 4) ^ … ^ (path[127].target << 4));

- 计算结果

  - 首先加上偏向权重*偏向系数

  bias_weights[bias_index]*bias_coefficient

  - 然后每 8 列加一次矢量乘积:

weights[index[i]][i*8...(i+1)*8]*coefficients[i*8...(i+1)*8]*h

$$h=\begin{cases}1 & (path[i].taken == true)\\ -1 & (path[i].taken == true)\end{cases}$$

- 如果结果 >= 0，表示预测跳转；反之，则表示不跳转

- 预测更新
  - 存储当前历史
  - 预测更新历史
  - 循环队列的插入方法

```
path[path_index] = PathInfo(actual, target);
path_index = (path_index + 1) % path_length;
```

训练策略

输入参数:predicted(预测结果), actual(实际跳转), ip

辅助参数:last_sum(预测值), last_bias_index, last_indexes(索引)，last_path(历史)，last_path_index(历史下标)

步骤:

- 计数器更新策略
  - 当预测不正确，计数器+1
  - 当预测正确且预测值小于阈值，计数器-1
- 阈值自适应策略
  - 当计数器为正数，阈值+1

- 当计数器为负数，阈值-1

- 权重更新策略

  - 当预测正确且预测值大于阈值，权重不更新

  - 反之，表示训练度不够，权重更新，利用预测时的索引去更新偏向权重和 128 个权重，如果历史位与实际跳转相同，表示关联度高，权重+1；反之，则权重-1

- 历史恢复

  - 当预测不正确，恢复历史

  - 更新历史

集成到 sniper 的一些改变

- 权重位数

  前:前 64 列权重取值范围为[-128, 127]，后 64 列[-64, 63]

  后:权重取值范围统一为[-128, 127]

- 小表行数

  前:第 1 小表行数为 512，后 15 个小表行数为 256

  后:小表行数统一为 512

- 循环队列存储历史

  前:更新历史通过拷贝

  后:更新历史通过下标，加快效率（采用训练时更新还能更加优化）

- 索引方法

前:位拼接

后:去掉后 4 位，异或

注：sniper 的预测器也是对于分支才进行预测