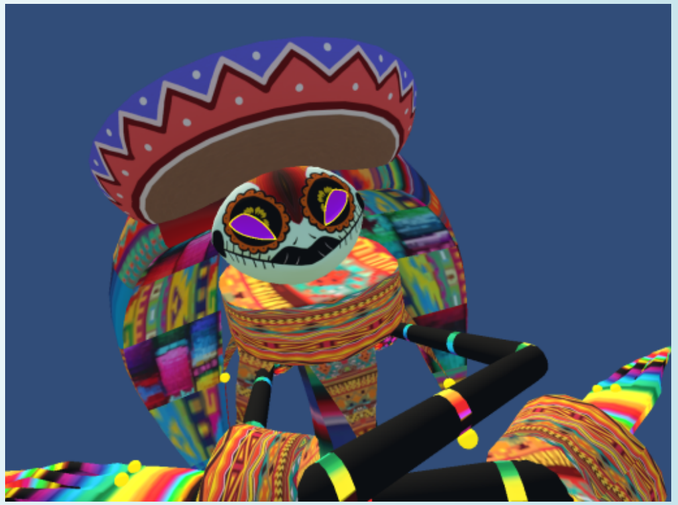
# 自作して見えた
# 感情分析モデルの限界

# 自己紹介

- Rin(rindguitar)
- 2024年11月からプログラミングを始めた
- 機械学習（NLP・時系列予測）に挑戦中
- GitHub
  https://github.com/rindguitar



# 今日のテーマ

1. 何を作っているのか？
2. レビューで学習とファインチューニングするだけで変わる？（OOD比較）
3. DAPTで汎用モデルに勝つ
4. その差は運じゃない？（多シード検証）
5. 感情分析の限界
6. まとめ — 学んだこと

# 1. 何を作っているか？

- Steam APIで英語レビューを集め、**感情分析**をするモデル
- ベースは **DistilBERT**。Steamレビュー（7ゲーム・計1万件）で学習
- 目的はレビューからゲーム需要予測に繋げること
- ラベルは `voted_up`（👍 / 👎）を流用 — 手軽だが**粗いproxy**（後で効いてくる）

# 学習と評価の設計

**学習（自作モデル）**

- モデルは **DistilBERT**（BERTの軽量版・約60%高速で性能97%維持 → 手元GPUで学習可）
- 入力はレビュー本文 + `voted_up`
- 1万件を train / val / test に分割
- 👍 / 👎 の2クラス分類として微調整 → loss低減

**評価（OOD設計）**

- 学習で使用した**7ゲームは除外**
- 未知の **20ゲーム・2000件**で測定
- 初見レビューで通用するか**汎化性能** を見る

# 2. ドメイン学習と
# ファインチューニングするだけで
# 変わる？

- **Steamレビュー学習の自作** vs **映画レビュー学習の汎用**（DistilBERT-Finetuned-SST-2）
  - 同じDistilBERTベースなので**ドメイン学習の効果だけ**を切り分けられる
- 判定は同じ2000件で両モデルを評価する **対応ありデータ** → **McNemar検定**（有意水準 **α=0.05**）

# 結果：ほぼ互角だった

| 指標 | 自作 (Steam) | 汎用 (映画) |
|---|---|---|
| Accuracy | **86.5%** | 85.1% |
| Precision | 83.5% | **87.7%** |
| Recall | **91.0%** | 81.6% |
| F1 | **87.1%** | 84.6% |

- Accuracyの差 +1.4pt は…
- **McNemar検定 p=0.105 → 有意差なし（誤差圏）**
- 学習とファインチューニングの付加価値は限定的

# 変わったのは「偏り」

- 変わったのは**スキル**ではなく**判定の偏り**
  - 自作 = **Recall寄り**（ポジと言いすぎ → FP多・皮肉に弱い）
  - 汎用 = **Precision寄り**（慎重）
- 土台のDistilBERTが性能の大半を担っていた

- 次の打ち手 → **DAPT**（ドメイン適応事前学習）で土台から鍛え直す

# 3. DAPTで汎用モデルに勝つ

- **DAPT = Domain-Adaptive Pre-Training**
- Steamレビューで**穴埋め問題（MLM）**を解かせる
  - 例：Great graphics, but the [MASK] loop gets repetitive fast. → gameplay
- ゲーム界隈の語彙・文脈を吸収 → その土台の上でファインチューニング

- **計10万件・572ゲーム**
  （OOD20＋学習7ゲームを除外＝リーケージ防止）

# 結果：汎用モデルに有意に勝利

**OOD（未知20ゲーム・2000件）**

| モデル | OOD Acc |
|---|---|
| **DAPT後** | **88.8%** |
| 旧自作 | 86.5% |
| 汎用(SST-2) | 85.1% |

**in-domain（1万件 test）**

| 指標 | DAPT前 | DAPT後 |
|---|---|---|
| Test Acc | 84.7% | **87.8%** |
| Train−Test gap | 12.8 | **7.4** |

過学習が縮小＝汎化向上

- McNemar：DAPT vs preDAPT **p=0.0001**
  DAPT vs sst-2 **p=0.0001**（ともに有意）

# DAPTは何を直したか

| | 直した | 壊した | 正味 |
|---|---|---|---|
| **FP** | 75 | 23 | **−52** |
| FN | 19 | 25 | +6 |
| 計 | 94 | 48 | **+46** |

純改善のほぼ全てが**FP削減**

- 直したのは対比・否定の長文（「**great … but terrible**」型）
- preDAPTが**ポジと誤読**していたのをDAPTが正した
- 代償として短文でわずかに過慎重化

# DAPTが正した実例（OODの実文）

**FP：褒めるが実は否定**

> *"**Beautiful artwork, but** I didn't find the gameplay loop engaging..."（Hades）*

DAPT前：ポジティブ ❌ →
DAPT後：ネガティブ ⭕

**FN：否定的な語に見えて称賛**

> *"It's actually a **really good** kart racer, the dev did a **fantastic job**..."（Nightmare Kart）*

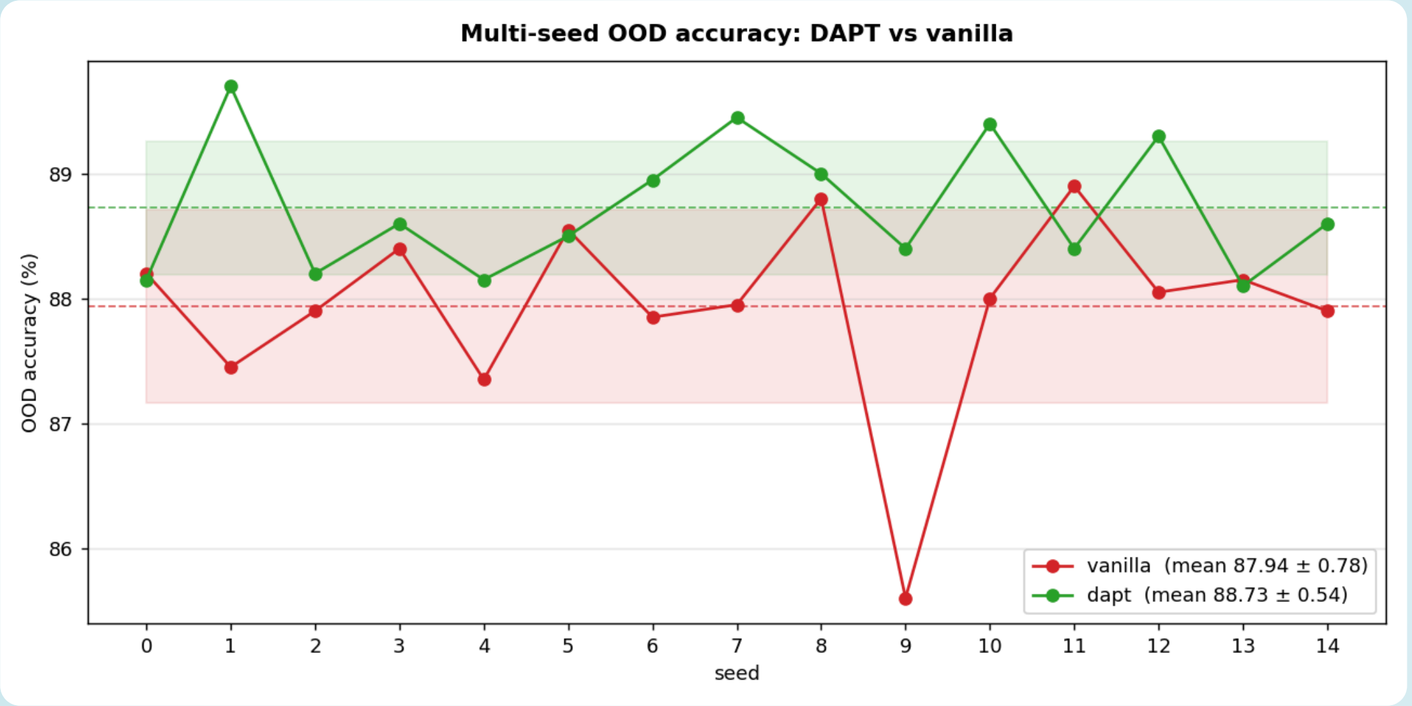
DAPT前：ネガティブ ❌ →
DAPT後：ポジティブ ⭕

# 4. その差は運じゃない？

- seed=42 の1回ずつでは、その差はシードの運かもしれない
- → **15回の独立試行**（seed 0〜14）で頑健性を確認（多シード反復）

- 気づき（学び）：**固定すべき乱数は2か所**あった
  - データ分割は固定していたが、**学習の内部**（重み初期化・shuffle・dropout）が抜けていた
  - ここを固定して初めて「各シード＝再現可能な1試行」になる

# 結果：+2.3PTは「盛れていた」




- 正直な効果量： **+0.79pt**
- 11/15シードでDAPT勝利
- 単発の +2.3pt は pre_dapt の**下振れ**で膨らんでいた

- 結論：**精度↑ ブレ↓**
  DAPTの効果は運ではなく
  **小さいが有意で安定**

# 5. 感情分析の限界

- DAPTで伸びたが、それでも **OOD Acc は約89%で頭打ち**
- 原因はモデルではなく **ラベルそのもの**
  - voted_up（👍 / 👎）は「本文の感情」ではなく「**推奨するか**」
  - 本文がネガでも👍、ポジでも👎 が混ざる
    → **ラベル自体がノイズ**

- つまり残りの誤りは、
  賢いモデルでも**消せない領域**に入っている

# 消せない誤り＝IRREDUCIBLE ERROR

**ラベルと本文がズレる例**

- 皮肉：「最高のバグ製造機 10/10」
  - 本文はネガ → でも 👍
- 両面評価：「神ゲーだが最適化が酷い」
  - 人によって 👍／👎 が割れる

- **正解ラベル自体が曖昧** → どれだけ鍛えても消えない（**Irreducible Error**）
- 人が見ても判定できない例が残る

- 「精度の天井」は性能不足とは限らず、**問題設定の限界**だった

# 6. まとめ ― 学んだこと

- **比較は公平に**：未知データ(OOD)＋対照実験で「微調整の価値」を切り分けた
- **DAPTは効く**：精度を安定させ、汎用モデルに有意に勝利。
- **差は統計で確かめる**：seed1発の数字を鵜呑みにせず、多シードで頑健性を検証
- **ラベルの限界**:精度、性能の問題と思っていたが、モデルを賢くしても消せない限界があった

# 参考リンク

- リポジトリ: game-demand-forecast
- 用語解説: GitHub Wiki（OOD / データリーケージ / McNemar検定 / DAPT ほか）