SNS 데이터, AI로 긁는 법 — 완전 재현 가이드

Threads·인스타 크롤링 + 반응률(ER) 분석 — Apify vs BrightData, 코드까지

> @junyoung.ai 무료 배포 리드 마그넷. **이 문서만 AI(Claude 등)에 주면 그대로 따라 해 동일한 결과를 낼 수 있도록** 실제 명령·actor ID·입력 포맷·스크립트를 다 넣었습니다. 우리가 실제로 만들어 쓰는 방식 그대로(2026-05 검증). 토큰만 본인 것으로 바꾸세요.

---

0. 이걸로 뭘 하나

- 내 분야 잘나가는 계정·글을 통째로 수집 → 본문·좋아요·댓글·리포스트·팔로워 확보
- **반응률(ER)** 로 "구조가 좋아 터진 글"을 가려냄(팔로워 무관)
- 좋은 글의 hook·구조·댓글체인을 역설계 → 내 콘텐츠에 적용
- 손으로 며칠 걸릴 일을 **30분, 커피값**에.

1. 도구 선택 (중요)

대상	도구	이유
Threads 글+댓글	**Apify**	BrightData엔 Threads 게시물·댓글 스크레이퍼가 없음(프로필만)
인스타·틱톡·링크드인	**BrightData**	dataset 풍부 + 대체로 더 저렴
Threads 프로필(팔로워만)	둘 다	—

2. 준비

```
# Apify: apify.com 가입 → Settings → API tokens
export APIFY_API_TOKEN="apify_api_xxx"       # 본인 토큰
# BrightData: brightdata.com → Account settings → API tokens
export BRIGHTDATA_API_TOKEN="xxxx-xxxx"      # 본인 토큰
# jq 설치 필요 (brew install jq / apt install jq)
```

3. Apify로 Threads 긁기

3-1. 비용 안전장치 — 건당 과금(PAY_PER_EVENT) actor만

```
# 실행 전 가격 모델 확인. PAY_PER_EVENT 아니면 쓰지 말 것(월구독형 과금 폭탄 방지)
curl -sS "https://api.apify.com/v2/acts/futurizerush~meta-threads-scraper?
```

```
token=$APIFY_API_TOKEN" \
  | jq '.data.currentPricingInfo.pricingModel'   # => "PAY_PER_EVENT" 여야 안전
```

3-2. 키워드로 Threads 글 검색 (작성자·반응·미디어 inline 반환)

actor: futurizerush/meta-threads-scraper (검색+프로필 데이터 한 번에)

```
cat > input.json <<'JSON'
{ "mode": "search", "search_keywords": ["1인기업","AI 자동화","n8n","솔로프리너"],
  "search_filter": "top", "results_limit": 40 }
JSON

# 동기 실행 → 결과 dataset 다운로드 (waitForFinish=300초)
RESP=$(curl -sS -X POST \
  "https://api.apify.com/v2/acts/futurizerush~meta-threads-scraper/runs?
token=$APIFY_API_TOKEN&waitForFinish=300" \
  -H "Content-Type: application/json" --data-binary @input.json)
DATASET=$(echo "$RESP" | jq -r '.data.defaultDatasetId')
echo "cost: $(echo "$RESP" | jq -r '.data.usageTotalUsd')"   # 실비용 확인
curl -sS "https://api.apify.com/v2/datasets/$DATASET/items?
token=$APIFY_API_TOKEN&clean=true&format=json" -o threads_posts.json
echo "saved $(jq length threads_posts.json) posts"
```

> 반환 필드: text_content, like_count, reply_count, repost_count, view_count, username, followers_count, post_url, has_media, media_type, media_url.

3-3. 특정 글의 댓글 체인 (작성자 자기 댓글 복원)

actor: futurizerush/threads-replies-scraper (입력은 **객체 배열**, max_replies ≤ 50)

```
cat > replies.json <<'JSON'
{ "post_urls": [{"url":"https://www.threads.com/@HANDLE/post/CODE"}],
  "max_replies": 50, "include_nested_replies": true }
JSON
RESP=$(curl -sS -X POST \
  "https://api.apify.com/v2/acts/futurizerush~threads-replies-scraper/runs?
token=$APIFY_API_TOKEN&waitForFinish=300" \
  -H "Content-Type: application/json" --data-binary @replies.json)
DS=$(echo "$RESP" | jq -r '.data.defaultDatasetId')
curl -sS "https://api.apify.com/v2/datasets/$DS/items?
token=$APIFY_API_TOKEN&clean=true" -o replies_out.json
# 작성자 본인 댓글만 (체인 = 글의 가치 페이로드)
jq -r '.[] | select((.author_username//"")=="HANDLE") | .text_content' replies_out.json
```

4. BrightData로 인스타 등 긁기

4-1. 쓸 dataset ID (API로 확인된 실제 값)

GET https://api.brightdata.com/datasets/list # 전체 목록(본인 계정)

대상	dataset_id
Instagram — Profiles	gd_l1vikfch901nx3by4
Instagram — Posts	gd_lk5ns7kz21pck8jpis
Instagram — Reels	gd_lyclm20il4r5helnj
Instagram — Comments	gd_ltppn085pokosxh13
Threads — Profiles	gd_mde7jg3ld2h3hnnf2
TikTok — Posts	gd_lu702nij2f790tmv9h
LinkedIn — People	gd_l1viktl72bvl7bjuj0

4-2. trigger → 폴링 → 다운로드 (Dataset API v3)

```
DATASET_ID="gd_l1vikfch901nx3by4"  # IG 프로필
echo '[{"url":"https://www.instagram.com/TARGET_HANDLE"}]' > bd_input.json

# 1) trigger → snapshot_id
SNAP=$(curl -sS -X POST \
  "https://api.brightdata.com/datasets/v3/trigger?
dataset_id=$DATASET_ID&include_errors=true" \
  -H "Authorization: Bearer $BRIGHTDATA_API_TOKEN" -H "Content-Type:
application/json" \
  --data-binary @bd_input.json | jq -r '.snapshot_id')

# 2) 완료까지 폴링
while :; do
  ST=$(curl -sS -H "Authorization: Bearer $BRIGHTDATA_API_TOKEN" \
       "https://api.brightdata.com/datasets/v3/progress/$SNAP" | jq -r '.status')
  echo "status=$ST"; [ "$ST" = "ready" ] && break; [ "$ST" = "failed" ] && exit 1; sleep 15
done

# 3) 다운로드
curl -sS -H "Authorization: Bearer $BRIGHTDATA_API_TOKEN" \
  "https://api.brightdata.com/datasets/v3/snapshot/$SNAP?format=json" -o ig_out.json
```

5. 반응률(ER) 랭킹 — 좋은 글 가려내기

```
import json, re
data = json.load(open("threads_posts.json", encoding="utf-8"))
def kr(s): return bool(re.search(r"[가-힣]", s or ""))
TOPIC = ("ai","에이전트","자동화","1인","사업","마케팅","퍼널","클로드","gpt","부업","프롬프트","창업")
rows, seen = [], set()
for p in data:
    t = p.get("text_content") or ""
    if not kr(t) or len(t) < 30: continue
    if not any(k in t.lower() for k in TOPIC): continue
    u = p.get("username"); f = p.get("followers_count") or 0
    L, R, RP = p.get("like_count") or 0, p.get("reply_count") or 0, p.get("repost_count") or 0
    if f < 300 or L < 20 or u in seen: continue        # 노이즈 컷
    er = (L + 2*R + 3*RP) / f                          # 반응률 = 좋아요+2*댓글+3*리포스트 / 팔로워
    rows.append((er, u, f, L, R, RP, t)); seen.add(u)
rows.sort(reverse=True)
for er,u,f,L,R,RP,t in rows[:10]:
    print(f"ER {er*100:.0f}%  @{u} (팔{f} ❤️{L} 💬{R} 🔁{RP})\n{t[:200]}\n---")
```

> 핵심: **절대 좋아요 ❌ → 팔로워 대비 반응률 ⭕**. 댓글·리포스트에 가중(2·3배). 팔로워 200짜리가 2만짜리를 이기는 게 흔하다 = *구조가 도달을 만든다*.

6. 꼭 챙길 것 (이게 진짜 알맹이)

1. **비용**: PAY_PER_EVENT actor만. 실행 응답의 usageTotalUsd로 실비 확인, 일일 캡 설정. 무료 크레딧(Apify $5/월) 의식. 대량·정기는 BrightData가 더 쌈.
2. **ban·윤리**: **수집만.** 자동 팔로우·댓글·DM = 계정 정지. 공개 데이터·약관·레이트리밋 존중.
3. **플랫폼 함정**: 인스타 핸들 ≠ Threads 핸들. 큰 IG 계정이 Threads엔 거의 없기도. **타깃 플랫폼에서 직접** 측정.
4. **분석 파라리시스**: 도구는 함정. 필요한 데이터만 딱 긁고 바로 쓴다.
5. **input 스키마**: actor/dataset마다 입력 필드 다름. 모르면 GET /v2/acts/<a>/builds/default(Apify) 또는 대시보드 API 탭(BrightData)에서 확인.

7. 전체 워크플로우 (end-to-end)

3-2 키워드 검색 → 5 ER 랭킹 → 상위 글 선별
→(체인 있으면) 3-3 댓글 수집
→좋은 글의 hook·구조 역설계 → 내 콘텐츠로 재작성

인스타 캐러셀로 배포할 때 (슬라이드)

1 SNS 데이터, 손으로 긁지 마세요 / 2 비용=건당 과금·캡 / 3 ban=자동 팔로우·댓글 금지 / 4 핸들 IG≠Threads / 5 도구 Threads=Apify·IG=BrightData / 6 좋아요 말고 반응률 / 7 전체 코드 무료(프로필 링크)

— 막히면 @junyoung.ai. 같이 봅니다.