Instructivo paso a paso: ejecución, extracción de resultados, lectura asistida por IA e interpretación de reportes

Genaro Carrasco Ozuna

Versión: v1.0.0 — DOI: 10.5281/zenodo.18049613

0. Enlaces operativos (1 clic)

- Página operativa (botones Colab): geozunac3536-jpg.github.io/TCDS_CENAPRED_COLLAP_REPO
- Repositorio GitHub: github.com/geozunac3536-jpg/TCDS_CENAPRED_COLLAP_REPO
- DOI (archivo permanente): doi.org/10.5281/zenodo.18049613
- Autor: ORCID 0009-0005-6358-9910 — Email: geozunac3536@gmail.com

1. Requisitos mínimos (sin instalación local)

1. Un navegador moderno (Chrome/Edge/Firefox).
2. Acceso a Google Colab (recomendado iniciar sesión con cuenta Google).
3. No se requiere instalar Python localmente.

2. Ejecución del sistema (AERC) y visualización de resultados

2.1 Abrir y ejecutar

1. Abra la página operativa.
2. Haga clic en **"Ejecutar sistema (AERC)"**. Se abrirá Colab con `colab/run_AERC.ipynb`.
3. En Colab, vaya a **Entorno de ejecución → Ejecutar todas**.
4. Espere a que terminen las celdas. El notebook:
   - instala dependencias desde `requirements.txt`,
   - descarga y ejecuta el script principal `code/code.py` (desde GitHub RAW),
   - genera archivos de salida en la sesión.

2.2 Dónde ver "los resultados" (dos niveles)

A) Resultados en pantalla (logs)
Durante la ejecución se muestran mensajes y estados en las celdas (salida estándar). Esto confirma que el flujo corrió.

B) Resultados como archivos (artefactos)
Los resultados se materializan como archivos dentro de la sesión Colab.

Cómo ubicarlos

1. En Colab, abra el panel izquierdo **Files** (icono de carpeta).
2. Busque archivos creados recientemente (ordenados por nombre o fecha).
3. Si existe una carpeta de salida (por ejemplo `outputs/` o `reports/`), ábrala.

Cómo abrirlos

- Archivos **.json / .jsonl**: click para previsualizar texto.
- Archivos **.png / .jpg**: click para ver imagen (gráficas).
- Archivos **.pdf**: click para abrir visor.
- Archivos **.gif**: click para reproducir secuencia/forense.

Cómo descargarlos

1. En **Files**, click derecho sobre el archivo → **Download**.
2. Alternativa: comprimir una carpeta y descargar:

```
# ejemplo si existe outputs/
!zip -r outputs.zip outputs
```

2.3 (Opcional) Guardar en Google Drive para no re-subir entre notebooks

Si desea evitar bajar/subir archivos entre pestañas:

1. En Colab, monte Drive:

```
from google.colab import drive
drive.mount('/content/drive')
```

1. Copie resultados a una carpeta de Drive (ejemplo):

```
!mkdir -p /content/drive/MyDrive/TCDS_results
!cp -r outputs/* /content/drive/MyDrive/TCDS_results/
```

3. Sellado criptográfico (SHA256) y verificación de integridad

3.1 Generar sello

1. Vuelva a la página operativa.
2. Haga clic en **"Generar sello (SHA256)"**. Se abrirá `colab/run_make_seal.ipynb`.
3. En Colab, suba el archivo resultado que desea sellar:
   - Panel **Files** → botón **Upload**, o
   - arrastrar y soltar en **Files**.
4. Ejecute las celdas. El proceso generará:
   - el hash SHA256 en pantalla,
   - y un archivo `<archivo>.sha256`.

3.2 Verificar sello

1. Vuelva a la página operativa.
2. Haga clic en **"Verificar integridad"**. Se abrirá `colab/run_verify_seal.ipynb`.
3. Suba **dos** archivos:
   - el archivo original (ej. `result.json`),
   - su sello (ej. `result.json.sha256`).
4. Ejecute las celdas. Resultado esperado:
   - **OK — INTEGRIDAD VERIFICADA** si coincide,
   - **FAIL — ARCHIVO MODIFICADO** si no coincide.

4. Qué documento contiene qué (mapa rápido de extracción)

Elemento	Dónde aparece	Qué extraer / para qué sirve
Notebook AERC	`colab/run_AERC.ipynb`	Logs de ejecución; ubicación/nombres de salidas; confirmación de corrida
Resultados (datos)	Files (Colab), típicamente `.json/.jsonl/.csv`	Valores numéricos, series temporales, métricas y reportes
Gráficas/figuras	Files (Colab), típicamente `.png/.pdf/.gif`	Evidencia visual del comportamiento y del flujo analítico
Sello SHA256	`<archivo>.sha256`	Huella criptográfica para demostrar integridad

Código principal	`code/code.py`	Fuente ejecutable auditada (lógica del pipeline)
Herramientas sello	`tools/make_seal.py`, `tools/verify_seal.py`	Sellado y verificación independiente
Licencia	`LICENSE.md` y `NOTICE.md`	Condiciones de uso, atribución y avisos

5. Lectura asistida por IA: prompts explícitos y cómo interpretar

5.1 Qué entregarle a la IA (paquete mínimo)

Para una lectura útil, adjuntar:

- 1 archivo de resultados (ej. `result.json` o el principal generado),
- 1–3 gráficas clave (PNG/PDF/GIF),
- (opcional) el sello `.sha256` si se desea trazabilidad.

5.2 Prompt recomendado (copiar/pegar)

```
Tarea: Interpreta estos resultados y gr ficas como revisor externo.
1) Resume qu  est  midiendo cada archivo/figura.
2) Identifica se ales fuertes vs ruido y posibles falsos positivos.
3) Se ala qu  evidencia ser a necesaria para confirmar o descartar la lectura.
4) Si hay m tricas/umbrales, explica si parecen conservadores o laxos.
5) Prop n 3 mejoras concretas al reporte para que sea m s auditable.
```

5.3 Cómo evitar lecturas engañosas (reglas simples)

- Si la IA afirma causalidad sin respaldo, pedir: **"¿qué evidencia exacta en el archivo sustenta eso?"**
- Si la IA "rellena" valores, pedir: **"cita el campo/clave exacta y el valor"**.
- Si la IA interpreta una gráfica sin metadatos, pedir: **"¿qué ejes, unidades y ventana temporal estás asumiendo?"**

6. Interpretación práctica de gráficas (guía corta, no técnica)

6.1 Lo que debe verificarse siempre

1. **Eje X**: ¿tiempo? ¿segundos/minutos? ¿ventana completa?
2. **Eje Y**: ¿amplitud cruda? ¿magnitud? ¿valor normalizado?
3. **Marcadores**: ¿hay una marca de "evento" o "t_C" (si aparece)?
4. **Comparación**: ¿la gráfica muestra antes/después o solo una porción?

6.2 Señales típicas y qué significan (lectura conservadora)

- **Picos aislados**: suelen ser ruido/transitorios si no hay persistencia.
- **Cambio sostenido de nivel**: puede indicar cambio de régimen (requiere contraste con base).
- **Patrones repetidos**: posible estructura, pero también artefacto (verificar estación/canal).
- **Gráfica "limpia" sin variación**: revisar si hubo datos suficientes o si hubo filtrado excesivo.

6.3 "Dónde se confirma" una lectura

Una interpretación es sólida si:

- se puede repetir la corrida en Colab y produce salidas consistentes,
- los archivos resultantes se pueden sellar y verificar,
- la gráfica corresponde a los datos (mismo rango temporal y misma fuente).

7. Dónde viven los falsos positivos (y cómo revisarlos)

1. Identifique en el resultado si existe:
   - nivel de "alerta", "estado" o "clasificación",
   - conteo de detecciones o disparos por ventana.
2. Compare contra un periodo cercano sin evento relevante (control):
   - si el sistema "dispara" igual, es ruido o umbral laxamente calibrado.
3. Pida a la IA (o revise manualmente): **"¿Qué condición exacta del código produjo el disparo?"**

8. Vías tecnológicas para disponer del método (de evaluación a adopción)

Este entregable está orientado a evaluación. Si una institución decide explorar adopción, vías típicas:

1. **Modo notebook (evaluación)**: ejecución en Colab y extracción manual de artefactos.
2. **Modo pipeline (CI/CD)**: ejecutar el script en runners (GitHub Actions / GitLab CI) con registros automáticos.
3. **Modo servicio (API)**: encapsular el flujo como servicio (Docker/REST) para integrar con tableros.
4. **Modo on-prem / soberano**: despliegue en infraestructura interna con control de datos y auditoría.
5. **Modo auditoría externa**: terceros ejecutan versiones fijadas por DOI, con sellado y reporte comparativo.

9. Checklist de evaluación (lo mínimo para una revisión justa)

- Ejecuté AERC en Colab sin modificar código.
- Localicé los artefactos en **Files** y los descargué.
- Generé un sello SHA256 y verifiqué integridad.
- Revisé al menos 1 resultado + 1 gráfica con lectura crítica.
- Identifiqué posibles falsos positivos y pedí evidencia exacta (campos/condiciones).
- Registré observaciones: qué funcionó, qué no, y qué haría falta para validar/descartar.

10. Contacto

- Genaro Carrasco Ozuna
- ORCID: 0009-0005-6358-9910
- Email: geozunac3536@gmail.com
- GitHub: geozunac3536-jpg