

✅ ¿Puedo hacer Early Fusion con modelos `Sequential`?

Situación	¿Funciona?	¿Por qué?
Los modelos son <code>Sequential</code> recién definidos	🟡 A veces	Funciona solo si no los has guardado/cargado, porque el grafo aún está en memoria
Los modelos son <code>Sequential</code> y cargados	❌ No	Keras pierde la conexión <code>input → capa interna</code> al cargar
Quieres usar <code>.get_layer('nombre')</code> + <code>Model(...)</code>	❌ No	<code>Sequential</code> no guarda la arquitectura funcional para eso
Usas <code>Sequential</code> sin extraer capas intermedias	✅ Sí	Pero eso no es Early Fusion real, es como un stack final

🧠 En resumen:

❌ No puedes hacer Early Fusion con dos modelos `Sequential` si necesitas acceder a capas internas.

✅ Sí puedes hacer Early Fusion si construyes los modelos con el API funcional.

✅ **Sí puedes hacer Early Fusion con modelos `Sequential` si NO los has cargado con `load_model()`**

Y solo si los has creado justo antes, en la misma ejecución.

🧠 ¿Por qué sí funciona si no los cargas?

Cuando defines un modelo `Sequential` y todavía no lo has guardado ni cargado, Keras:

- Mantiene el grafo completo de conexiones en memoria,
- Puedes acceder a su `.input` y `.get_layer(...).output`,
- Y por tanto, puedes construir un nuevo modelo con `Model(inputs=..., outputs=...)` sin errores.

👉 En ese caso, tu código sí funcionará:

```
python                                                                    Copiar Editar

tabular_intermediate = Model(
    inputs=tabular_model.input,
    outputs=tabular_model.get_layer('intermediate_layer_1').output
)
```

Siempre que `tabular_model` haya sido creado con `Sequential(...)` justo antes.

🚫 Pero ojo: esto es frágil

En cuanto hagas esto:

```
python                                                                    Copiar Editar

tabular_model.save("modelo_tabular.keras")
tabular_model = load_model("modelo_tabular.keras")
```

❌ Dejará de funcionar ese `Model(inputs=..., outputs=...)` con `.get_layer(...)`.

Porque el grafo ya no está reconstruido de forma funcional.

✅ En resumen

Caso	¿Funciona en Early Fusion?
Modelo <code>Sequential</code> recién creado	✅ Sí
Modelo <code>Sequential</code> cargado con <code>.h5</code>	❌ No
Modelo funcional (<code>Model(inputs, outputs)</code>)	✅ Siempre

✅ Ese bloque que me pasas **es funcional**, y **debe seguir siéndolo** para que Early Fusion funcione bien.

Y no, no puedes convertirlo a `Sequential` para "arreglar" nada, porque:

🔥 **No puedes usar `Sequential` para construir un modelo como este, porque:**

1. `Sequential` solo acepta una única entrada (y tú tienes dos: `tabular_input` y `vision_input`).
2. `Sequential` no permite bifurcaciones ni fusiones de capas (como tu `Concatenate()`).
3. `Sequential` no permite reusar partes intermedias de otros modelos de forma explícita.

📌 **Entonces, ¿por qué ese código funciona (o no)?**

✅ Ese bloque funciona si:


- `tabular_model` y `vision_model` han sido creados en modo funcional o secuencial justo antes,
- No los has guardado/cargado con `load_model()`,
- Las capas `'intermediate_layer_1'` y `'our-fc1'` existen en el modelo y están accesibles.

❌ Ese bloque no funciona si:

- Has guardado y cargado `tabular_model` o `vision_model` como `Sequential`,
- Y luego intentas hacer `Model(inputs=..., outputs=una_capa_intermedia)` como haces en `tabular_intermediate` o `vision_intermediate`.

Conclusión clara

 Ese modelo de fusión tiene que ser funcional, sí o sí.

 No se puede convertir a `Sequential`. No es una limitación tuya: es una limitación de Keras.

























Y para que funcione sin errores:

- Asegúrate de que `tabular_model` y `vision_model` son funcionales o recién creados secuenciales (y no cargados),
- O reescríbelos con `Model(inputs, outputs)` si vas a guardarlos, cargarlos y luego reusarlos en fusiones.

Depende de lo que necesites.

Pero en general, si vas a trabajar con modelos un poco más complejos o reutilizables, como en *Early Fusion*, es mejor usar el API funcional.

Comparativa: `Sequential` vs `Functional`

Característica	<code>Sequential</code>	<code>Functional</code> (<code>Model(inputs, outputs)</code>)
 Fácil de usar	 Muy fácil: capas en orden	 Requiere definir entradas y salidas
 Para modelos lineales (1 entrada → 1 salida)	 Perfecto	 También sirve
 Varios inputs o outputs	 No se puede	 Sí
 Reusar capas o ramas	 No	 Sí
 Extraer submodelos o capas intermedias	 Muy limitado (solo recién creados)	 Perfecto
 Control sobre el grafo	 No, es oculto	 Total
 Guardar y cargar para reusar partes	 Problemático	 Seguro
 Ideal para Early Fusion	 No (salvo casos muy específicos)	 Sí, es lo suyo



💡 Entonces, ¿cuándo usar cada uno?

👉 Usa `Sequential` si:

- Tu modelo es **súper simple y lineal** (tipo: input → Dense → Dense → output).
- No vas a **guardar y reusar partes**.
- Solo necesitas **una entrada y una salida**.

👉 Usa `Functional` si:

- Tienes más de **una entrada o una salida**.
- Necesitas **fusionar datos** de distintos tipos (como en *Early Fusion*).
- Vas a **extraer capas intermedias**, guardar modelos y reusarlos.
- Quieres **flexibilidad total y robustez a futuro**.

🔄 Resumen rápido para ti

Si estás aprendiendo o prototipando algo simple: usa `Sequential`.

Pero si vas en serio (como en tu caso con fusión temprana):

🔥 Usa el API funcional, y no lo sueltes nunca más.

¿Quieres que te muestre un ejemplo de cómo pasar un modelo `Sequential` que ya tienes a funcional, conservando todo igual?