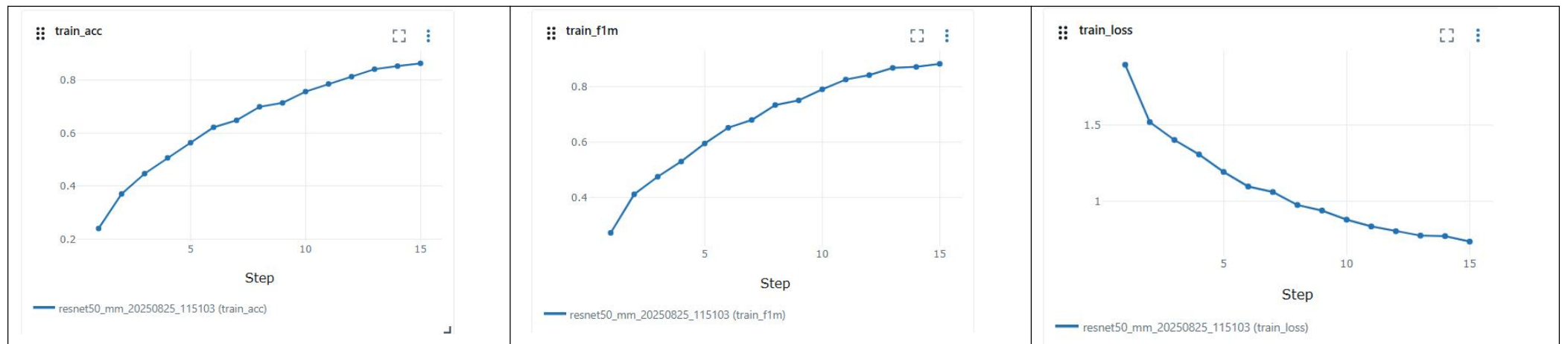


Hiperparámetros

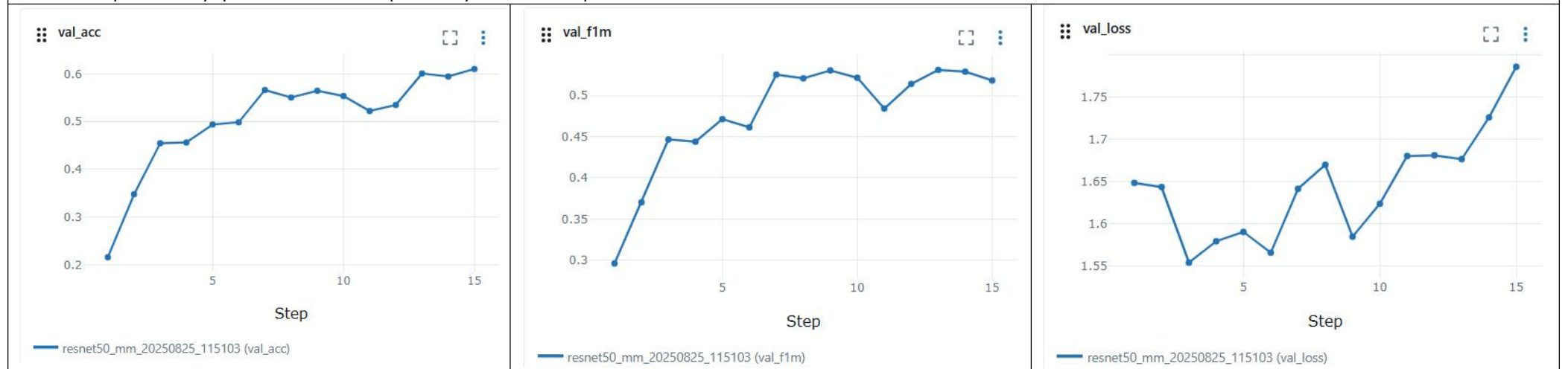
BATCH_SIZE = 32, EPOCHS = 15, LR = 1e-4, WEIGHT_DECAY = 2e-4

train_Sofia2--- Modelo: resnet50 --- Tiempo 3.9h

	TRAIN	VAL	TEST	
ACC	0.8637	0.6101	0.6436	Vemos un Overfitting
LOSS	0.7351	1.7859	1.1664	Como la perdida es más del doble se confirma el Overfitting
F1	0.8816	0.5183	0.5872	El modelo no generaliza bien las clases minoritarias



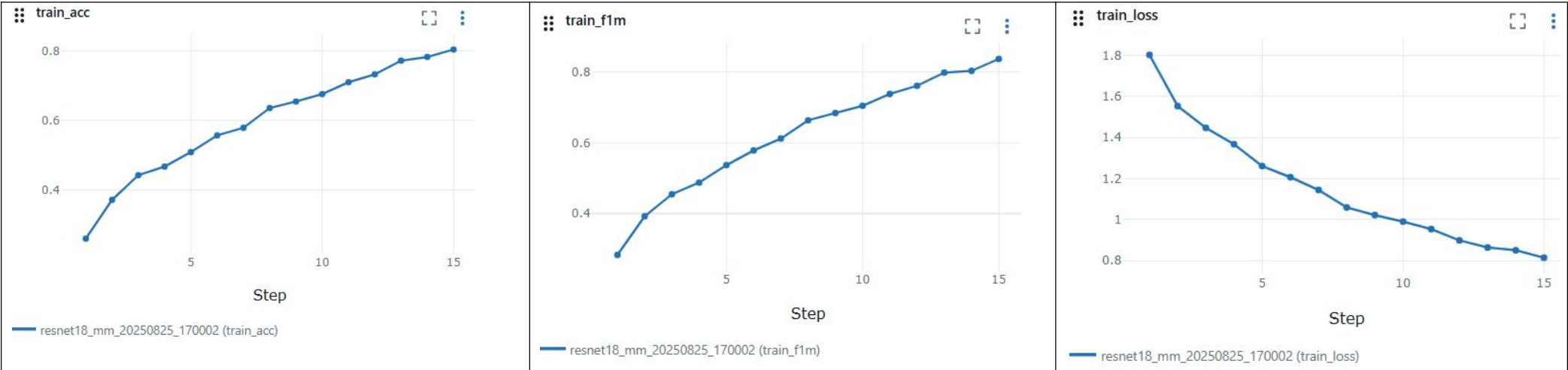
El modelo aprende muy rpaido a minimizar la prerdida y maximizar la precisión



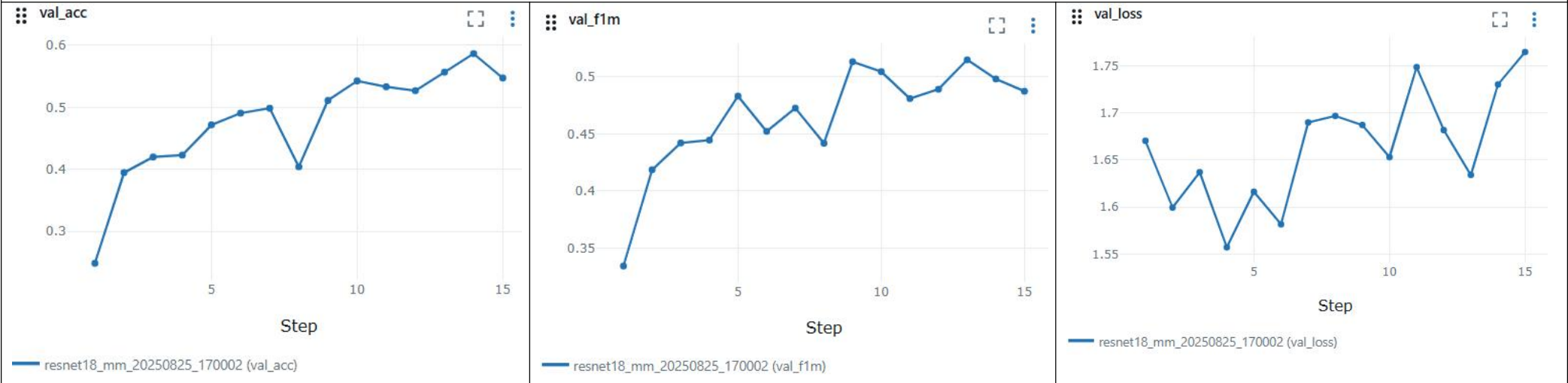
El punto optimo se encuentra en el epoch entre 8-10, después el modelo empezo a generalizar peor.

train_Sofia7--- Modelo: resnet18 --- Tiempo 1.3h

	TRAIN	VAL	TEST	
ACC	0.8042	0.5472	0.5824	Vemos un Overfitting
LOSS	0.8140	1.7647	1.1465	Como la perdida es más del doble se confirma el Overfitting
F1	0.8369	0.4872	0.5390	Observamos un mal rendimiento. Entrena muy rápido pero no logra generalizar bien.



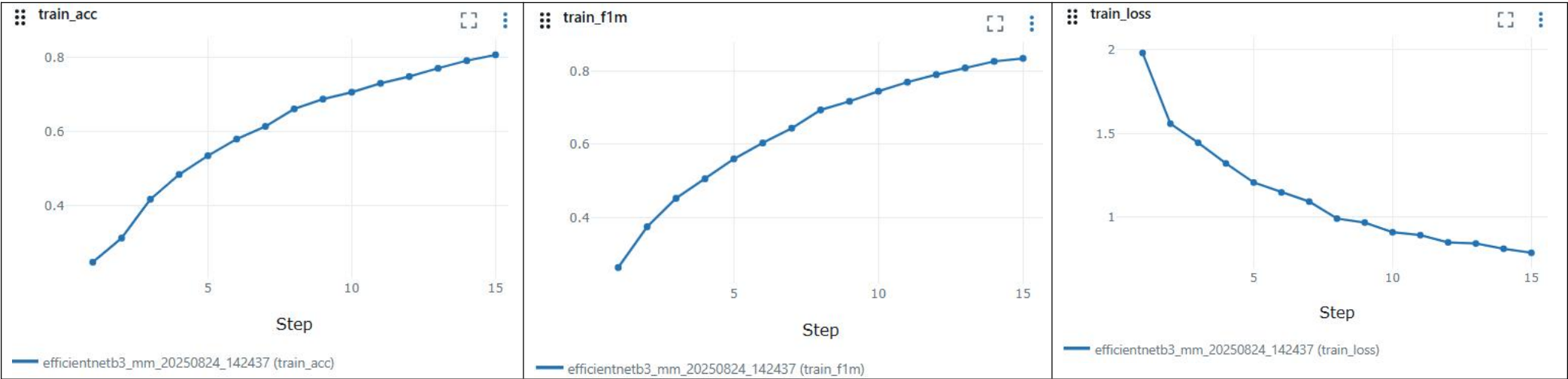
Las curvan son suaves y consistentes. Tiene una buena capacidad de ajuste a los datos en el entrenamiento



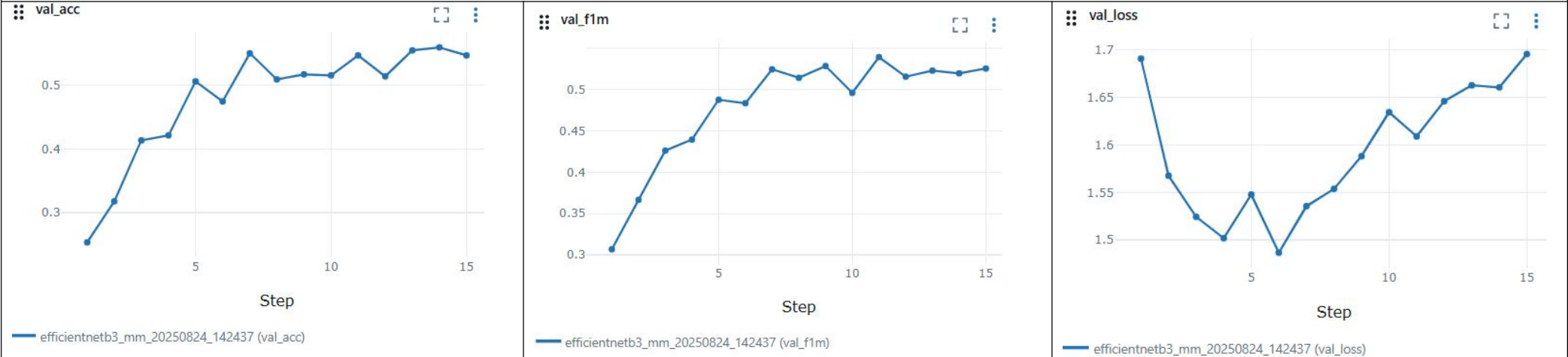
Observamos curvas muy volátiles, sobre todo en la perdida. ES muy inestable.

train_Sofia3--- Modelo: efficientnet_b3 --- Tiempo 3.3h

	TRAIN	VAL	TEST	
ACC	0.8075	0.5472	0.6075	Vemos un Overfitting, no tan fuerte como el anterior.
LOSS	0.7867	1.6955	1.0749	Como la perdida es más del doble se confirma el Overfitting
F1	0.8350	0.5253	0.5725	El modelo no generaliza bien las clases minoritarias



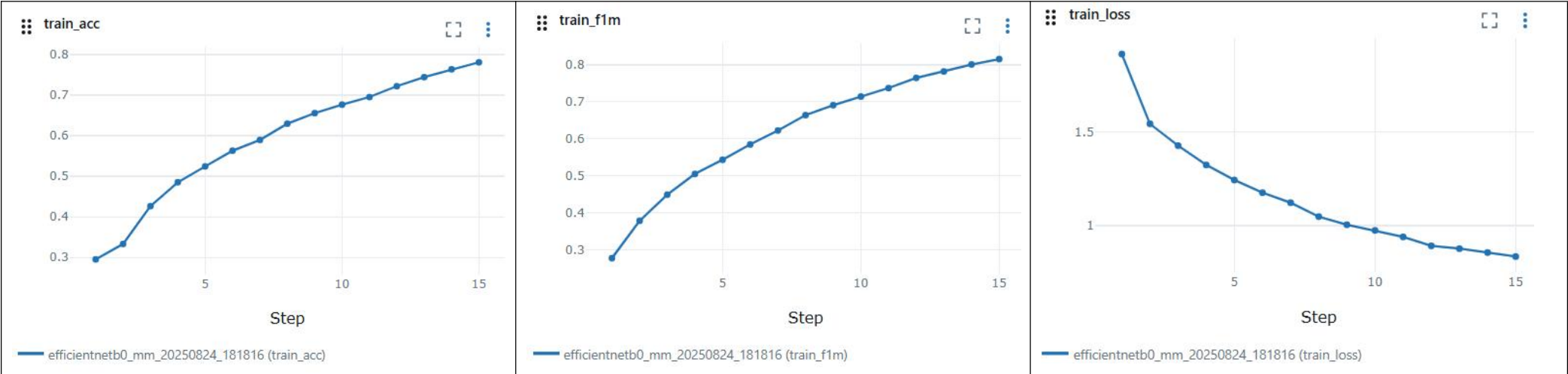
El modelo continua mejorando a lo largo de los 15 epochs



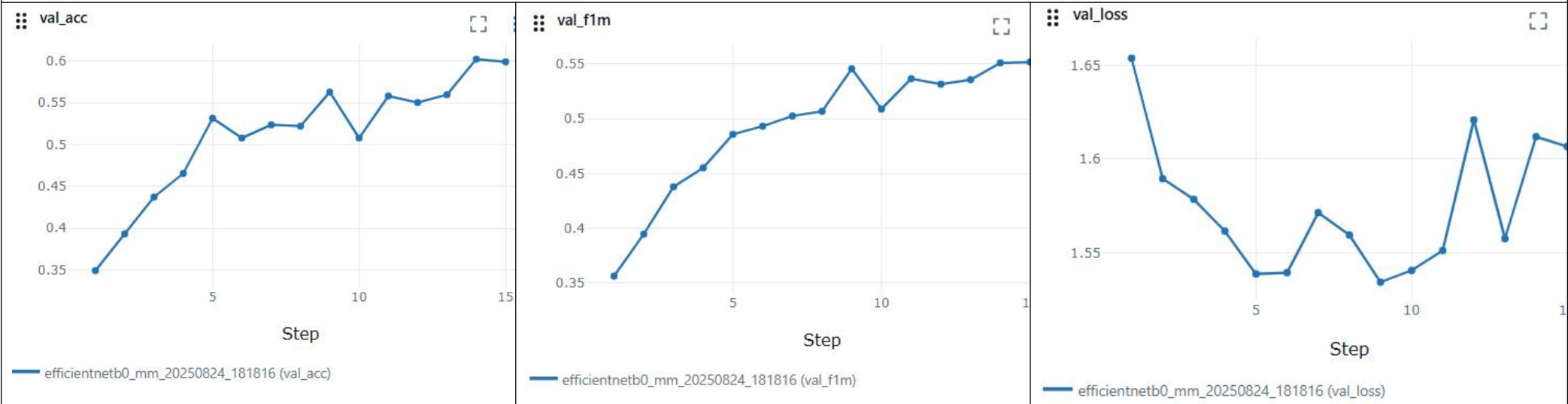
El epoch 6 alcanza el punto más bajo de perdida. A partir de este observamos que empieza a subir hasta terminar muy alto.

train_Sofia4 --- Modelo: efficientnet_b0 --- Tiempo 1.4h

	TRAIN	VAL	TEST	
ACC	0.7808	0.5991	0.6185	Vemos un Overfitting, no tan fuerte como el anterior.
LOSS	0.8341	1.6066	1.1233	Se confirma el Overfitting
F1	0.8156	0.5518	0.6008	Obtenemos una de las mejor métrica, obteniendo un mejor rendimiento de generalización



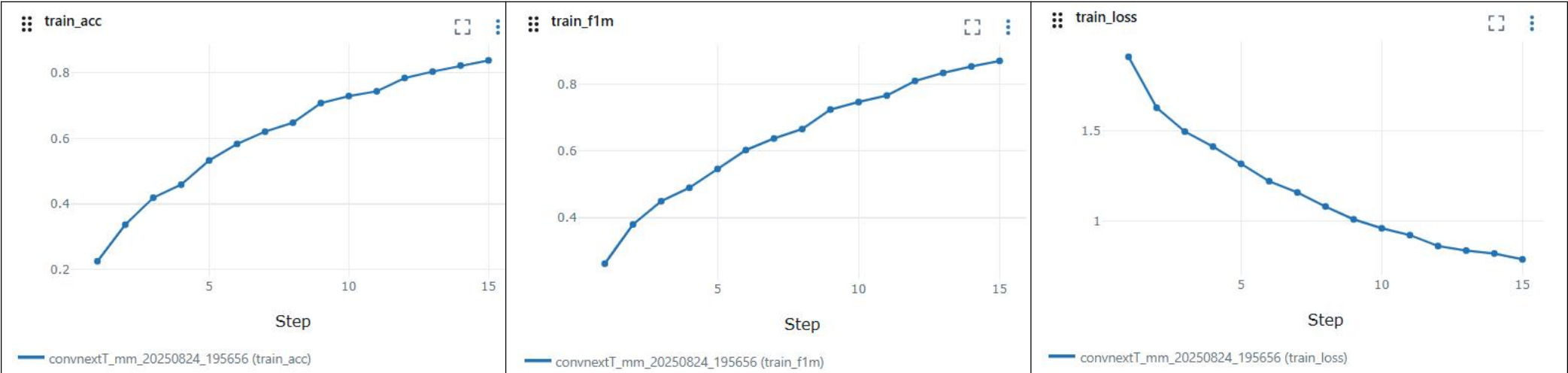
Las curvas parece que van estables comparado con el efficientnet_b3



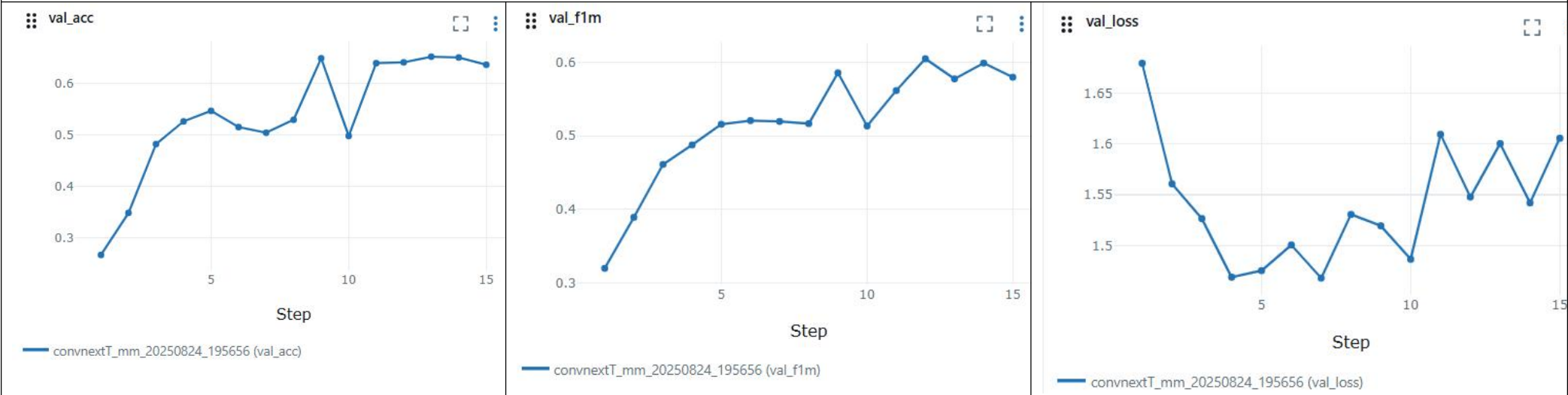
El mejor momento es alrededor del epoch 10-12 para todas métricas. En loss despues del epoch 12 empieza a subir drásticamente.

train_Sofia5 --- Modelo: convnext_tiny --- Tiempo 3.4h

	TRAIN	VAL	TEST	
ACC	0.8383	0.6368	0.6311	Vemos un Overfitting, similar al EfficientNet-B3
LOSS	0.7863	1.6057	1.0848	Se confirma el Overfitting
F1	0.8688	0.5797	0.6081	Obtenemos la mejor métrica, obtiene el mejor rendimiento de generalización pero a un costo de entrenamiento bastante mayor que EfficientNet-B0



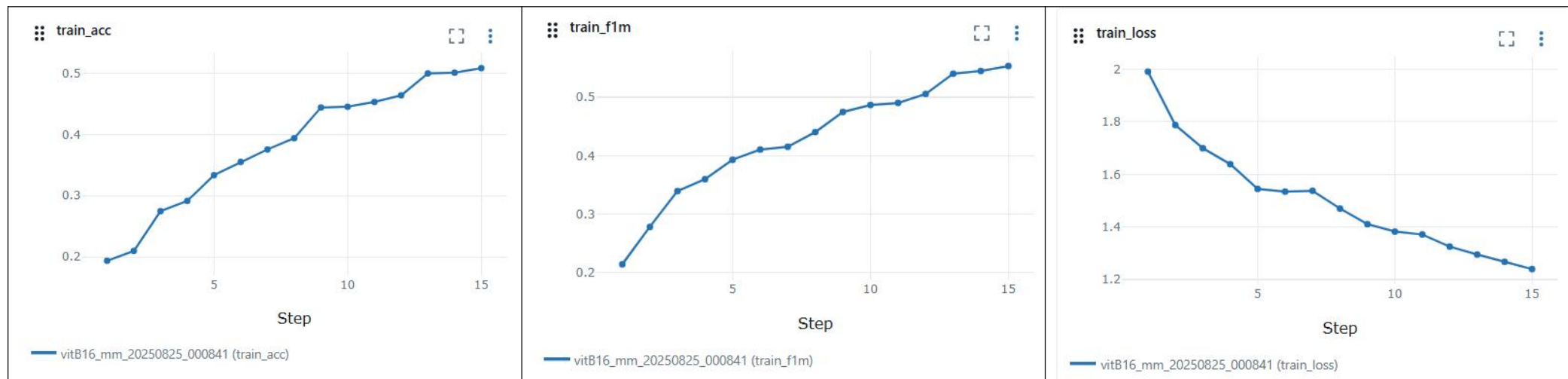
Observamos un entrenamiento suave y sin estancarse, con curvas consistentes.



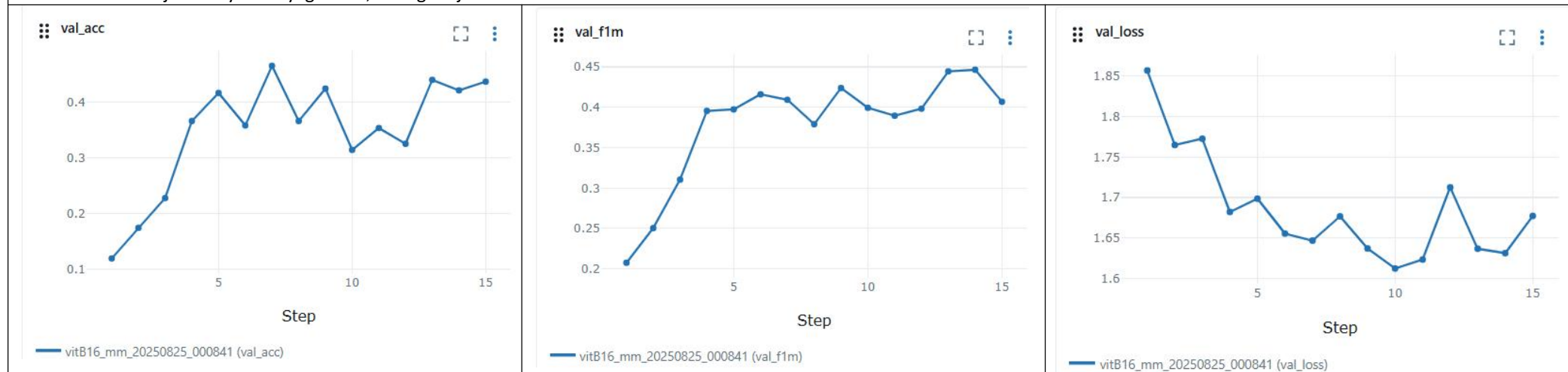
Podemos ver que alrededor del epoch 10 encontramos el mejor rendimiento. Pero la mejor perdida la obtenemos alrededor del epoch 6. Podemos observar mucha inestabilidad para encontrar un mínimo estable para generalizar.

train_Sofia6 --- Modelo: vit_b_16 --- Tiempo 9.9h

	TRAIN	VAL	TEST	
ACC	0.5080	0.4371	0.4333	Tiene una precisión muy baja.
LOSS	1.2397	1.6772	1.2075	La perdida en validación es bastante más alta que en train
F1	0.5530	0.4068	0.4767	Tenemos el peor rendimiento de generalización de todos.



Muestran una mejora muy lenta y gradual, no logra ajustarse correctamente a los datos.

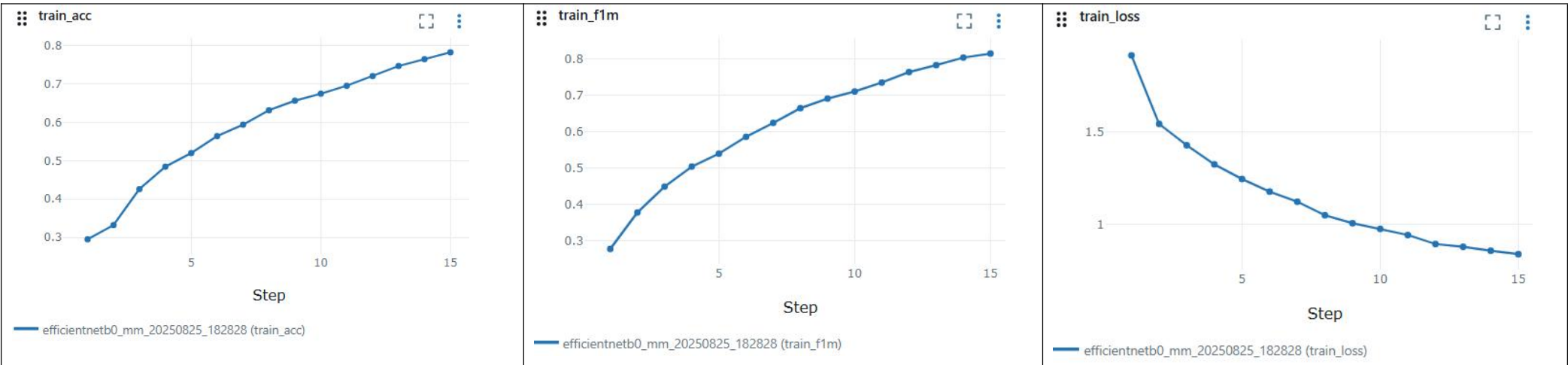


Observamos muchísimas fluctuación, la perdida nunca llega a converger.

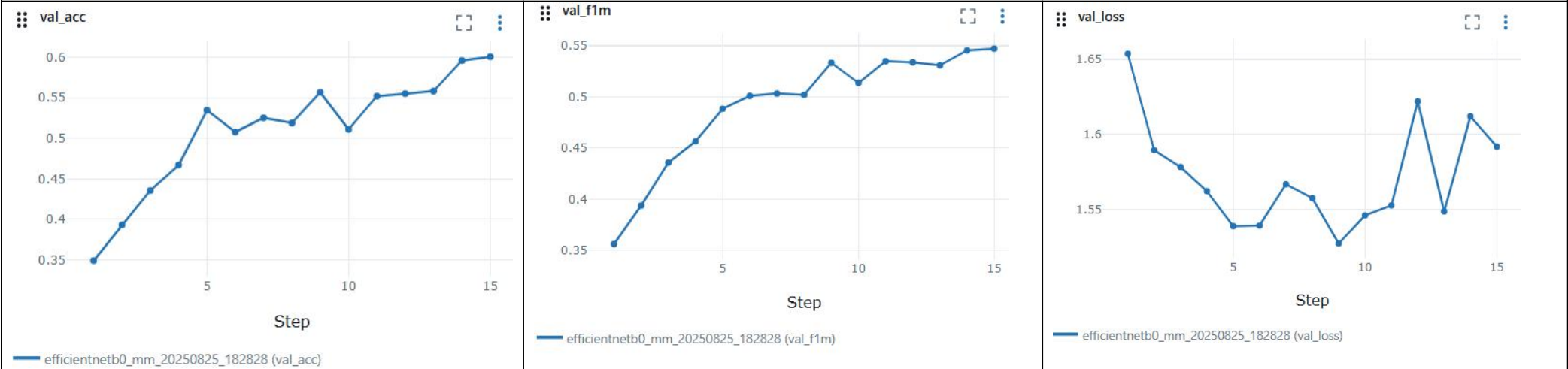
train_Sofia4_2--- Modelo: efficientnet_b0 --- Tiempo 1.1h

Cambios WEIGHT_DECAY = 5e-4 y patience = 4

	TRAIN	VAL	TEST	
ACC	0.7814	0.6006	0.6358	Su supero el valor en Test.
LOSS	0.8355	1.5919	1.1151	La perdida en test es más baja.
F1	0.8150	0.5471	0.6133	Nos indica que tenemos una mejora en la clasificación en las clases minoritarias



Las curvas son suaves y consistentes, demostrando que el modelo va aprendiendo.

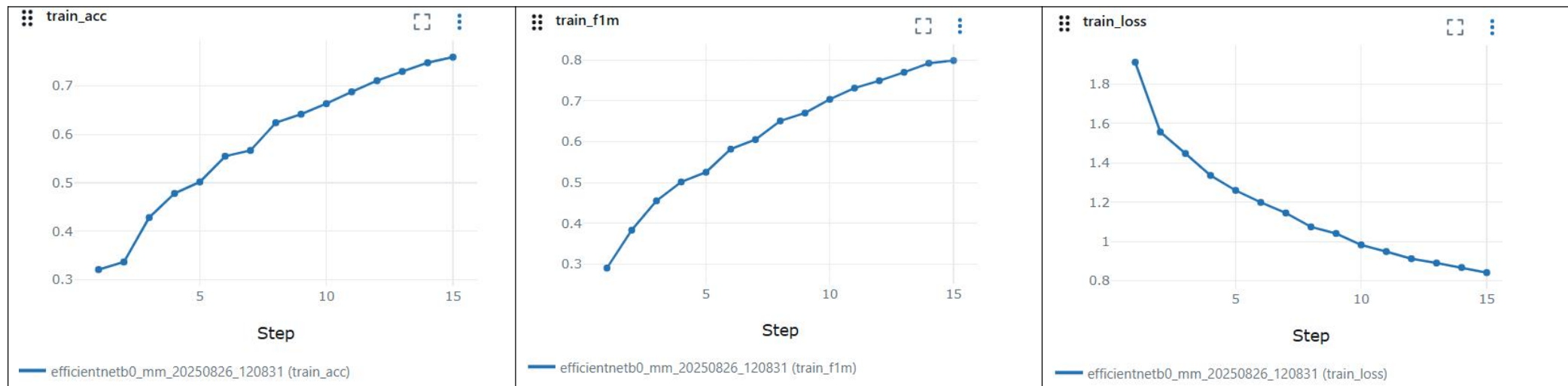


Parece que el momento óptimo es por el epoch 6, observamos menos fluctuaciones.

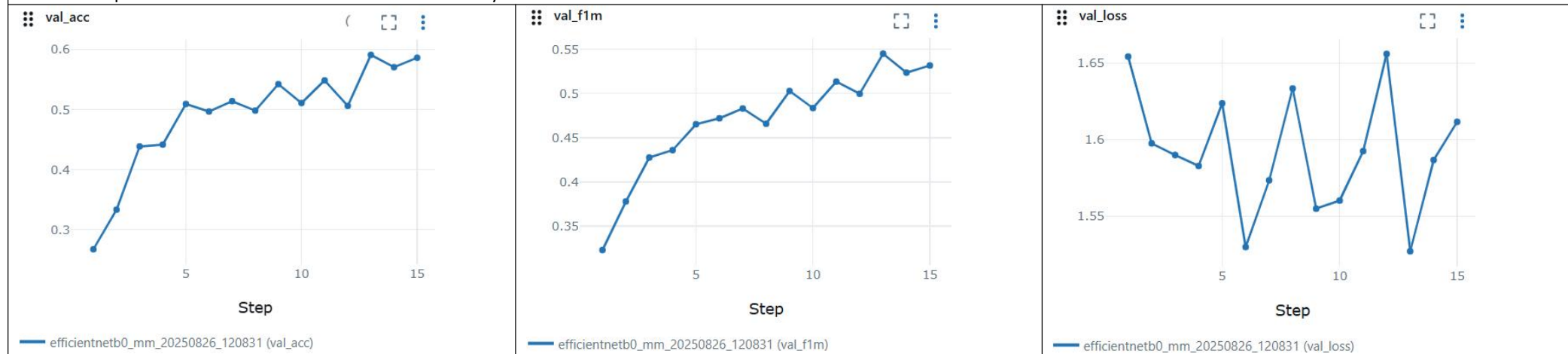
train_Sofia4_3--- Modelo: efficientnet_b0 --- Tiempo 1.0h ---- Con el Data Augmentation nuevo

Con WEIGHT_DECAY = $5e-4$ y patience = 4

	TRAIN	VAL	TEST	
ACC	0.7592	0.5865	0.6358	Es mejor en test que en val pero sigue siendo significativamente menor que train.
LOSS	0.8420	1.6117	1.0973	Podemos observar un sobreajuste.
F1	0.7987	0.5318	0.5913	Considero que el modelo no generaliza bien los datos nuevos.



El modelo aprende bien en el train. Las curvas son suaves y constantes.

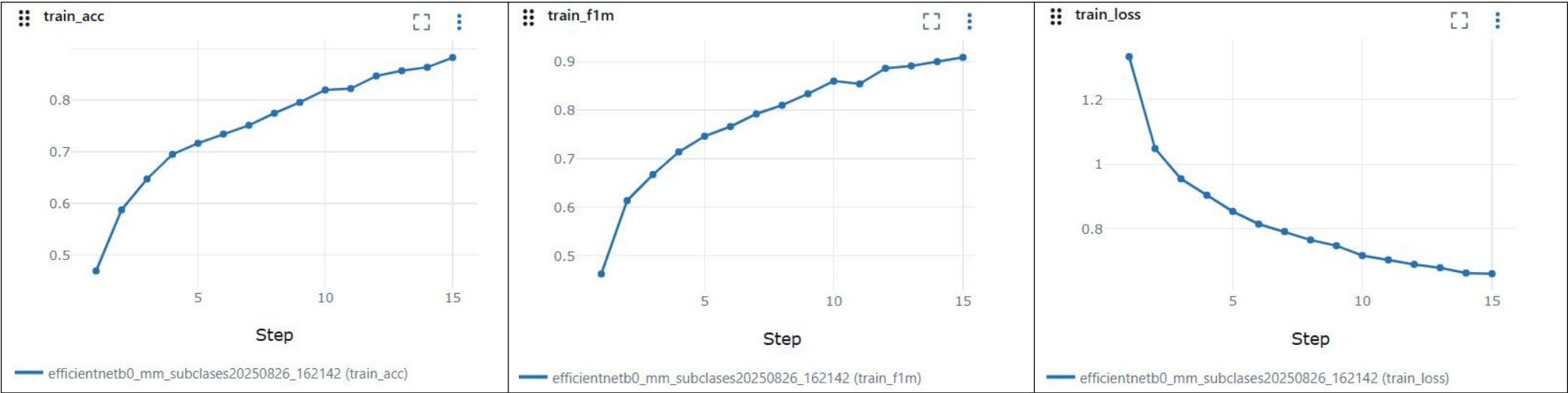


Tiene dificultades para generalizar. La perdida es muy volátil, la precisión se estanca y va fluctuando sin mejora clara.

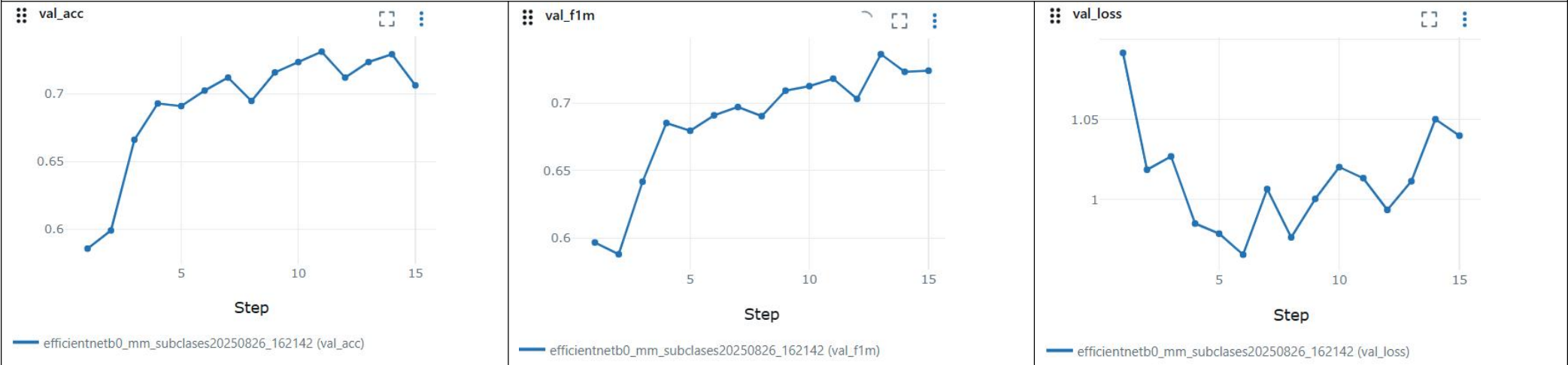
train_Sofia4_4--- Modelo: efficientnet_b0 --- Tiempo 53.8min ---- Con el Data Augmentation nuevo y subclases

Con WEIGHT_DECAY = 5e-4 y patience = 4

	TRAIN	VAL	TEST	
ACC	0.8825	0.7061	0.7543	El mejor valor en Test, mostrando un mejor sobreajuste.
LOSS	0.6606	1.0399	0.6446	Una perdida notablemente más baja, el modelo empieza a generalizar
F1	0.9087	0.7240	0.7771	Podemos ver que el modelo esta clasificando mejor las clases minoritarias



Curvas constantes y suaves, muestran un aprendizaje estable y efectivo.



Observamos algunas fluctuaciones pero en general tiene tendencia a la mejora.

train_Sofia4_5--- Modelo: efficientnet_b0 --- Tiempo 1.4 h ---- Sin el Data Augmentation nuevo y con subclases

Con WEIGHT_DECAY = 5e-4 y patience = 4

	TRAIN	VAL	TEST	
ACC	0.9578	0.7271	0.7524	En train tenemos una precisión excelente, pero en las otras se estanco más abajo.
LOSS	0.5786	1.1732	0.6963	Podemos observar un sobreajuste.
F1	0.9667	0.7362	0.7723	En train fue excelente y de igual forma que ACC con las otra se estanco más abajo.

