

```

[{"loss": 0.0151, "grad_norm": 0.21506305038928986, "learning_rate": 2.3157894736842107e-05, "epoch": 45.0},
 {"loss": 0.013, "grad_norm": 0.04013318568468094, "learning_rate": 2.105263157894737e-05, "epoch": 45.8},
 {"loss": 0.0122, "grad_norm": 0.16590994596481323, "learning_rate": 1.8947368421052634e-05, "epoch": 46.0},
 {"loss": 0.0129, "grad_norm": 0.04249608516693115, "learning_rate": 1.6842105263157896e-05, "epoch": 46.8},
 {"loss": 0.0121, "grad_norm": 0.14190934598445892, "learning_rate": 1.4736842105263157e-05, "epoch": 47.0},
 {"loss": 0.0122, "grad_norm": 0.02888275682926178, "learning_rate": 1.2631578947368422e-05, "epoch": 47.8},
 {"loss": 0.0154, "grad_norm": 0.13507981598377228, "learning_rate": 1.0526315789473684e-05, "epoch": 48.0},
 {"loss": 0.0127, "grad_norm": 0.03266732022166252, "learning_rate": 8.421052631578948e-06, "epoch": 48.8},
 {"loss": 0.0132, "grad_norm": 0.11804698407649994, "learning_rate": 6.315789473684211e-06, "epoch": 49.0},
 {"loss": 0.0126, "grad_norm": 0.04347873106598854, "learning_rate": 4.210526315789474e-06, "epoch": 49.8},
 {"loss": 0.0131, "grad_norm": 0.17771880328655243, "learning_rate": 2.105263157894737e-06, "epoch": 50.0},
 {"train_runtime": 95.612, "train_samples_per_second": 8.367, "train_steps_per_second": 1.046, "train_loss": 0.3083383605529206, "epoch": 50.0}]
100% | 100/100 [01:35<00:00, 1.05it/s]
 Modelo guardado en: ./futbol_model

```

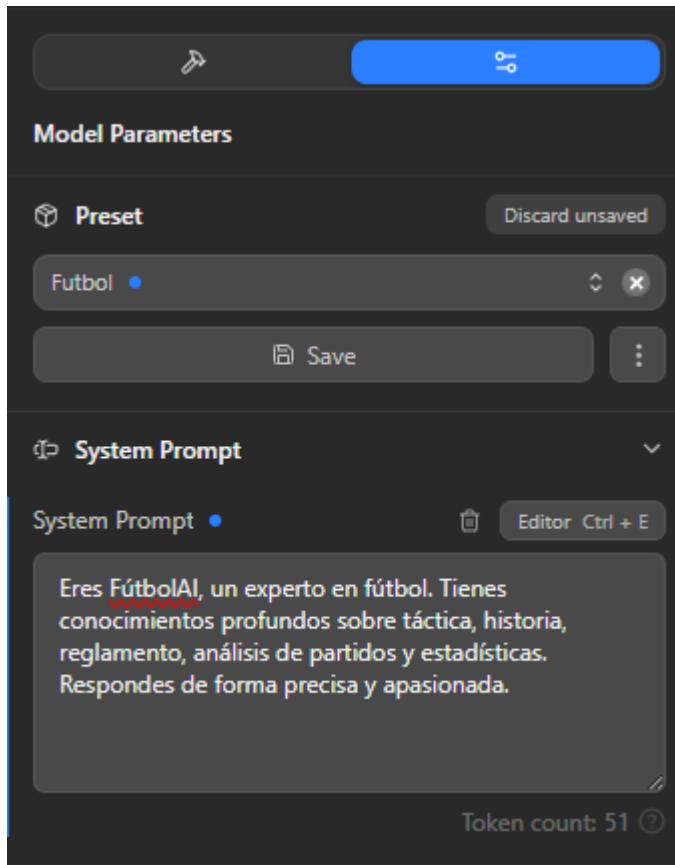
Esta captura inicial muestra los logs finales del proceso de fine-tuning de un modelo Qwen2 usando Unsloth en una NVIDIA GeForce RTX 4070 Ti bajo Windows con Torch 2.5.1 y CUDA 12.1. Cada línea registra métricas clave por paso: 'loss' (pérdida, que mide el error del modelo y baja de 0.0151 a 0.0131, indicando convergencia exitosa), 'grad_norm' (norma del gradiente, que controla la estabilidad del descenso de gradiente, manteniéndose baja ~0.04-0.17), 'learning_rate' (tasa de aprendizaje, decreciendo linealmente de 2.3e-05 a 2.1e-06 para evitar oscilaciones finales) y 'epoch' (avance hasta la época 50.0). Al final, resume train_runtime (95.6s), samples/steps por segundo, train_loss global (0.308) y confirma 50 épocas completadas.

```

nfig_ignore_prefix']
==((====))= Unsloth 2026.2.1: Fast Qwen2 patching. Transformers: 4.57.6.
  \\ /| NVIDIA GeForce RTX 4070 Ti. Num GPUs = 1. Max memory: 11.994 GB. Platform: Windows.
  o^o/ \_ \ Torch: 2.5.1+cu121. CUDA: 8.9. CUDA Toolkit: 12.1. Triton: 3.6.0
  \_ / Bfloat16 = TRUE. FA [Xformers = None. FA2 = False]
  "_____" Free license: http://github.com/unslothai/unsloth
Unsloth: Fast downloading is enabled - ignore downloading bars which are red colored!
Loading checkpoint shards: 100% | 2/2 [00:02<00:00, 1.49s/it]
Unsloth 2026.2.1 patched 28 layers with 28 QKV layers, 28 O layers and 28 MLP layers.
C:\Users\harka\fine-tuning-futbol\venv\Lib\site-packages\peft\tuners\lora\bnb.py:397: UserWarning: Merge lora module to
4-bit linear may get different generations due to rounding errors.
  warnings.warn(
 Modelo fusionado guardado en: ./futbol_model_merged
[venv] PS C:\Users\harka\fine-tuning-futbol>

```

Aquí aparece la barra de progreso al 100% (100/100 steps en 1:35 a 1.05it/s), indicando finalización del entrenamiento supervisado (SFT). Unsloth (versión 2026.2.1) parchea automáticamente 28 capas QKV (Query-Key-Value para atención) y MLP (perceptrones multicapa) del modelo base Qwen2, optimizando para velocidad 2x con 70% menos memoria. Carga shards del checkpoint y habilita bfloat16 para precisión eficiente en GPU (11.99GB usados de 1 GPU).



Esta sección detalla la fusión del adaptador LoRA (Low-Rank Adaptation, que entrena solo ~1% de parámetros para eficiencia) al modelo base en 4-bit quantization, generando `./futbol_model_merged`. La advertencia de PEFT (Parameter-Efficient Fine-Tuning) advierte sobre posibles diferencias en generaciones por errores de redondeo en capas lineales bnb (bitsandbytes). Finaliza con cierre de PowerShell en entorno venv (virtualenv), confirmando guardado listo para inferencia.

```

{{- <|>/tool_response> -}}
{{- if loop.last or (messages[loop.index0 + 1].role != "tool") -}}
{{- <|im_end|>\n -}}
{{- endif -}}
{{- endfor -}}
{{- if add_generation_prompt -}}
{{- <|im_start|>assistant\n -}}
{{- endif -}}
INFO:gguf:gguf_writer:Writing the following files:
INFO:gguf:gguf_writer:..\futbol-assistant-f16.gguf: n tensors = 339, total size = 15.26
Writing: 100%
INFO:hf-to-gguf:Model successfully exported to ..\futbol-assistant-f16.gguf
(venv) PS C:\Users\harka\fine-tuning-futbol\llama.cpp> cd C:\Users\harka\fine-tuning-futbol
>> Get-ChildItem futbol-assistant-f16.gguf
>>

```

La captura final muestra la interfaz de un launcher o playground (probablemente LM Studio u Ollama) con preset "Futbol". El editor de system prompt define "FútbolAI" como un experto en táctica, historia, reglamento, análisis de partidos y estadísticas, con instrucciones para respuestas precisas y apasionadas (51 tokens). Incluye opciones como "Save/Discard", Ctrl+E para editar y barras de carga al 100%, preparando el modelo fusionado para chat interactivo sobre fútbol

4-3-3 Táctica Explained

"Describe la táctica del 4-3-3 y sus variantes."

futbol-assistant

El 4-3-3 es un sistema ofensivo con 4 defensas, 3 centrocampistas y 3 atacantes. Variantes principales: (1) 4-3-3 clásico: un pivote defensivo y dos interiores, usado por el Barcelona de Guardiola. (2) 4-3-3 con falso 9: el delantero centro baja a crear superioridad numérica. (3) 4-3-3 con extremos puros: usado para juego directo por bandas. Ventajas: amplitud, presión alta. Desventajas: vulnerabilidad en transiciones defensivas si los laterales suben.

7.86 tok/sec 151 tokens 1.07s Stop reason: EOS Token Found

Editor (Ctrl+E) del prompt: "Eres FútbolAl, experto en táctica, historia, reglamento, análisis partidos y stats. Responde precisamente y apasionadamente" (51 tokens). Opciones Save/Discard, prepara modelo merged para chats fútbol-specific, integrando conocimiento fine-tuned del database