

# Informe Técnico: Variaciones en la Entrada y Salida de Modelos de IA

## 1. Campo `outputConfig`: Propósito y Funcionamiento

El campo `pack_format` en la configuración JSON de un modelo de AA define la forma estructural en la que el modelo devuelve sus salidas (`raw_output`) luego de la inferencia. Establece un contrato para saber de antemano qué tipo de desempaquetado debe aplicar antes de ejecutar el adaptador de salida (`output_adapter`).

### Posibles Valores:

Valor	Descripción
<code>raw</code>	Entrada en formato IR del sistema.
<code>boxes_scores</code>	Entrada en forma de ( <code>boxes</code> , <code>class_scores</code> ) o ( <code>class_scores</code> , <code>boxes</code> ) donde: <ul style="list-style-type: none"><li><code>boxes</code>: <math>(N,4) \rightarrow [ymin, xmin, ymax, xmax]</math></li><li><code>class_scores</code>: <math>(N,C)</math></li></ul>
<code>tflite_de_tpost</code>	Entrada en forma de ( <code>boxes</code> , <code>scores</code> , <code>classes</code> [, <code>count</code> ]) donde: <ul style="list-style-type: none"><li><code>boxes</code>: <math>(1,N,4)</math> o <math>(N,4) \rightarrow [ymin,xmin,ymax,xmax]</math></li><li><code>scores</code>: <math>(1,N)</math> o <math>(N, )</math></li><li><code>classes</code>: <math>(1,N)</math> o <math>(N, )</math></li><li><code>count</code>: <math>(1, ) \rightarrow</math> opcional</li></ul>
<code>anchor_deltas</code>	Entrada en forma de ( <code>box_deltas</code> , <code>class_scores</code> ) o ( <code>class_scores</code> , <code>box_deltas</code> ) donde: <ul style="list-style-type: none"><li><code>box_deltas</code>: <math>(1,N,4)</math> o <math>(N,4) \rightarrow [ty,tx,th,tw]</math></li><li><code>class_scores</code>: <math>(1,N,C)</math> o <math>(N,C) \rightarrow</math> logits o probas</li></ul>

`yolo_flat` Entrada en forma de tensor (N, 5+C) con columnas [cx,cy,w,h,obj,p0...pC]

---

## 2. Informe General de Variaciones desde Inputs hasta Outputs

### Sección `inputConfing`

- `width`: Ancho en píxeles requerido por el modelo.
- `height`: Alto en píxeles.
- `channels`: Cantidad de canales (1 para escala de grises, 3 para color RGB/BGR).
- `color_order`: Define el orden de los canales: "RGB" o "BGR".
- `normalize`: Aplica normalización por canal.
- `mean` y `std`: Valores por canal para normalizar (ej. Imagenet).
- `scale`: Divide los valores por 255 si es `true`.
- `letterbox`: Si se aplica `resize` + `padding` para conservar aspecto.
- `auto_pad_color`: Color del relleno.
- `preserve_aspect_ratio`: Mantiene proporciones originales.

### Subestructura `inputTensorConfing`

- `layout`: Representación del tensor. Valores:
  - `HWC`, `CHW`, `NHWC`, `NCHW`
- `dtype`: Tipo de dato esperado: `float32`, `uint8`, `int8`.
- `quantized`: Si el modelo está cuantizado.

### Sección `output`

- `confidence_threshold`: Umbral para descartar predicciones con baja confianza.
- `apply_nms`: Si se aplica supresión de no-máximos (NMS).
- `nms_threshold`: Umbral de IoU para el NMS.
- `nms_per_class`: especifica una ruta de NMS distinta.
- `top_k`: valor específico de `non_max_supresion`.
- `apply_conf_filter`: especifica una ruta de scores distinta.
- `pack_format`: estructura ya explicada.

## Subestructura `tensor_structure`

- `box_format`: Formato de las coordenadas de salida. Valores:
  - `xyxy`, `cxcywh`, `yxyx`
- `coordinates`: Diccionario con los índices de las coordenadas.
- `confidence_index`: Índice donde se encuentra la confianza.
- `class_index`: Índice donde se encuentra el ID de clase.
- `num_classes`: Total de clases detectables.

Este documento estandariza todos los posibles contratos con un modelo de AA utilizable por el sistema.