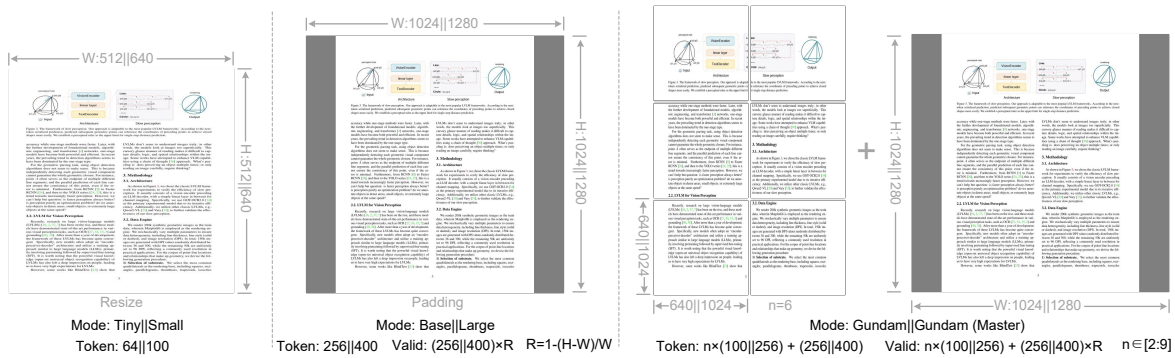

Figure 4 | To test model performance under different compression ratios (requiring different numbers of vision tokens) and enhance the practicality of DeepSeek-OCR, we configure it with multiple resolution modes.

the 4096 tokens go through the compression module and the token count becomes 4096/16=256, thus making the overall activation memory controllable.

Table 1 | Multi resolution support of DeepEncoder. For both research and application purposes, we design DeepEncoder with diverse native resolution and dynamic resolution modes.

Mode	Native Resolution				Dynamic Resolution	
	Tiny	Small	Base	Large	Gundam	Gundam-M
Resolution	512	640	1024	1280	640+1024	1024+1280
Tokens	64	100	256	400	n×100+256	n×256+400
Process	resize	resize	padding	padding	resize + padding	resize + padding

3.2.2. Multiple resolution support

Suppose we have an image with 1000 optical characters and we want to test how many vision tokens are needed for decoding. This requires the model to support a variable number of vision tokens. That is to say the DeepEncoder needs to support multiple resolutions.

We meet the requirement aforementioned through dynamic interpolation of positional encodings, and design several resolution modes for simultaneous model training to achieve the capability of a single DeepSeek-OCR model supporting multiple resolutions. As shown in Figure 4, DeepEncoder mainly supports two major input modes: native resolution and dynamic resolution. Each of them contains multiple sub-modes.

Native resolution supports four sub-modes: Tiny, Small, Base, and Large, with corresponding resolutions and token counts of 512×512 (64), 640×640 (100), 1024×1024 (256), and 1280×1280 (400) respectively. Since Tiny and Small modes have relatively small resolutions, to avoid wasting vision tokens, images are processed by directly resizing the original shape. For Base and Large modes, in order to preserve the original image aspect ratio, images are padded to the corresponding size. After padding, the number of valid vision tokens is less than the actual number of vision tokens, with the calculation formula being:

$$N_{valid} = \lceil N_{actual} \times [1 - ((max(w,h) - min(w,h))/(max(w,h)))] \rceil \quad (1)$$

where w and h represent the width and height of the original input image.