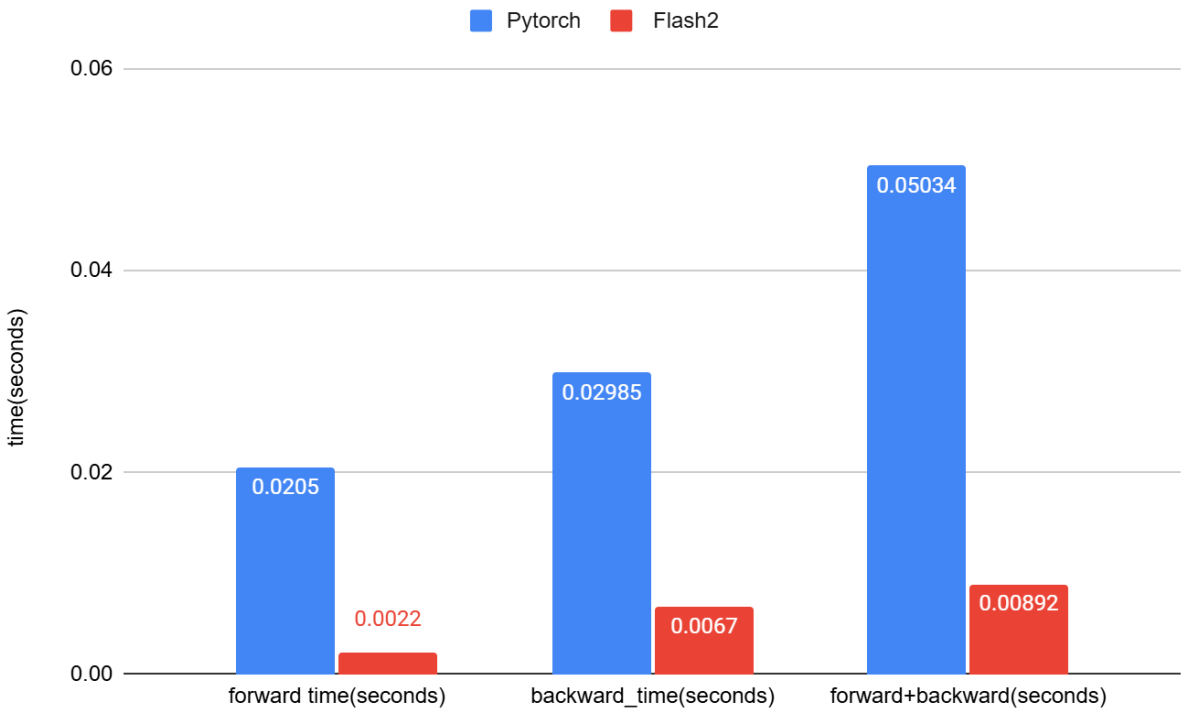
## 1. 比較 PyTorch 和 FlashAttention v2 的性能

### 運行時間(time(s))

- FlashAttention v2 在所有參數實驗中均顯示出顯著較低的前向（forward）、反向（backward）和前後向（forward_backward）運行時間。
- **固定參數:** Batch size = 16 , seq_len = 1024 , num_heads = 32 , emb_dim = 2048

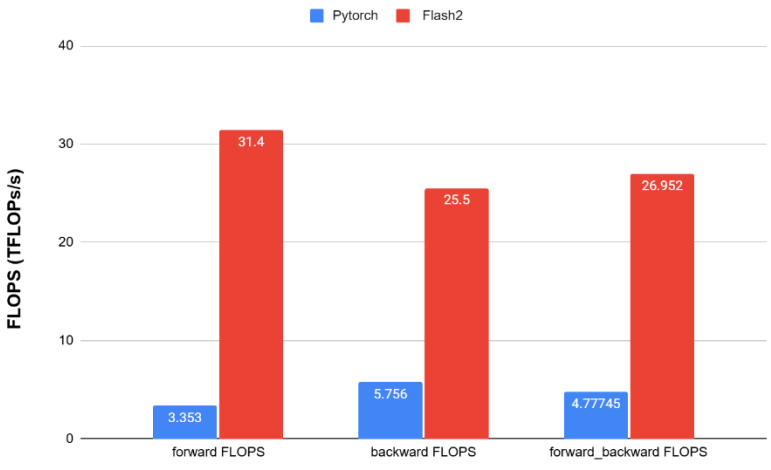
  FlashAttention V2 的 forward 傳播速度提高了 9 倍，backward 傳播速度提高了約 4.5 倍




### FLOPS(TFLOPs/s)

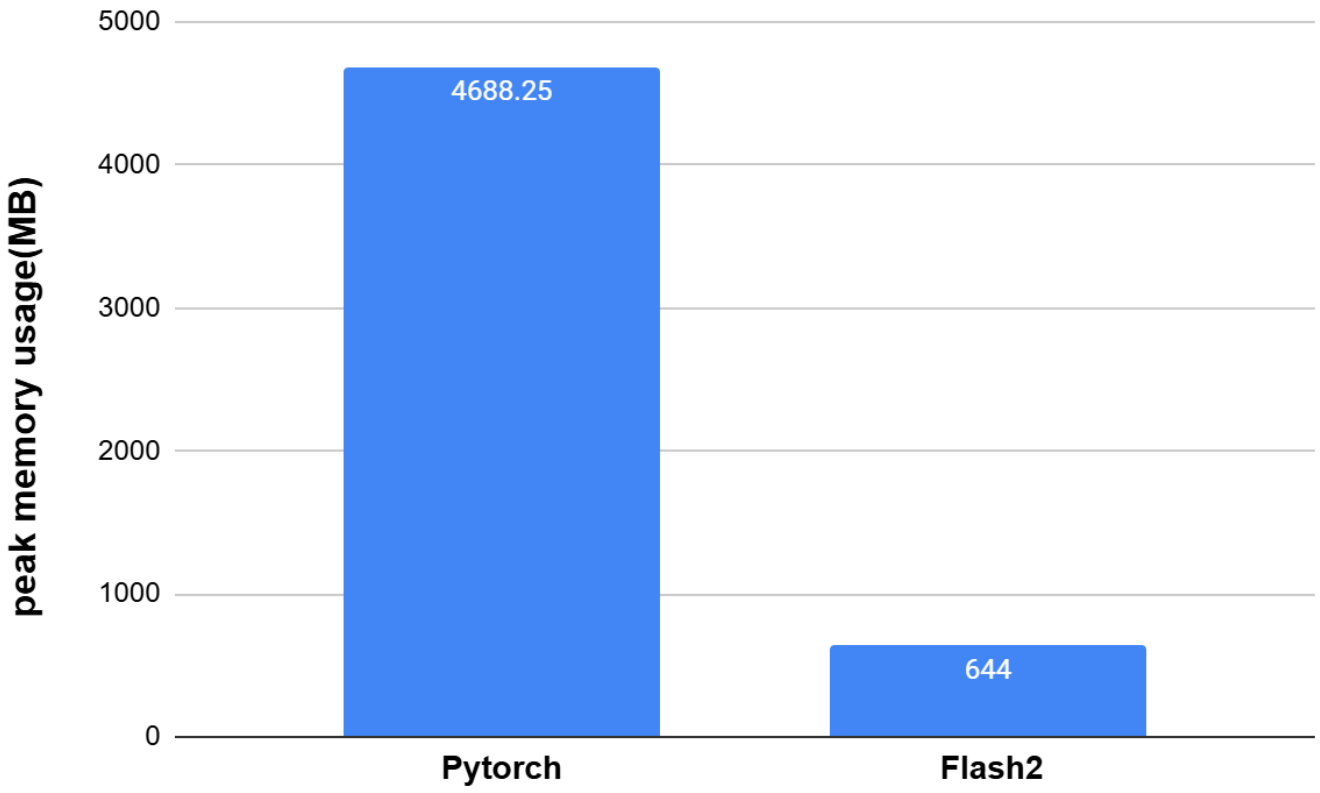
- FlashAttention V2 在所有參數實驗中均顯示出 FLOPS 明顯高於 PyTorch，表示其實現對硬體資源的利用效率更高。
- **固定參數:** Batch size = 16 , seq_len = 1024 , num_heads = 32 , emb_dim = 2048

  **Flash2 的 Flops 值在 forward 階段為 Pytorch 的 10 倍，backward 階段為 Pytorch 的 5 倍**



記憶體使用 **peak_memory_usage(MB)**

- 在所有參數實驗中均顯示 FlashAttention 的峰值記憶體使用量（peak memory usage）大幅低於 PyTorch，節約了 **50% 到 90%** 的記憶體。對於記憶體受限的設備（如 GPU），這種節約對於實現大模型的訓練和推理至關重要。
- **固定參數:** Batch size = 16 , seq_len = 1024 , num_heads = 32 , emb_dim = 2048
- **Flash2 記憶體用量僅 Pytorch 的 14%**




---

**2. FlashAttention 的優勢分析**

計算效率

- FlashAttention 通過優化的計算路徑和內部張量操作，有效減少了不必要的計算。
- 其利用了低精度數據類型（如 float16），並結合專為 GPU 設計的高效計算內核，提升了計算效能。

因果注意力（**Causal Attention**）的處理

- FlashAttention 在處理長序列和高維嵌入向量時對因果掩碼（causal mask）的操作更加高效，相較 PyTorch 有更好的擴展性。

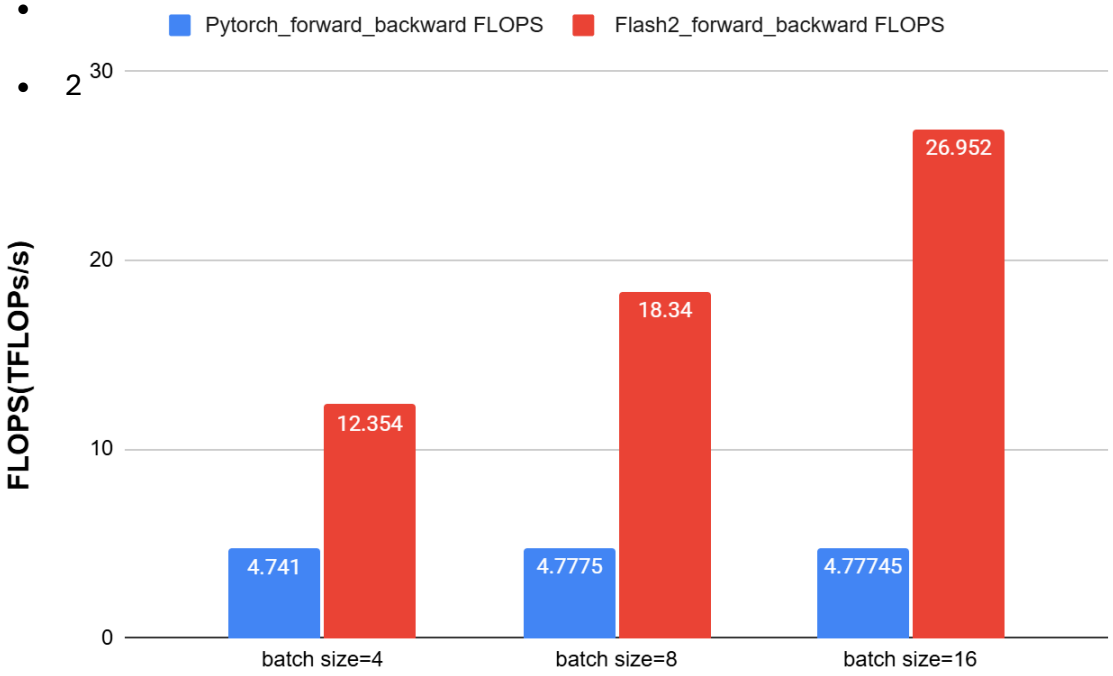
記憶體占用

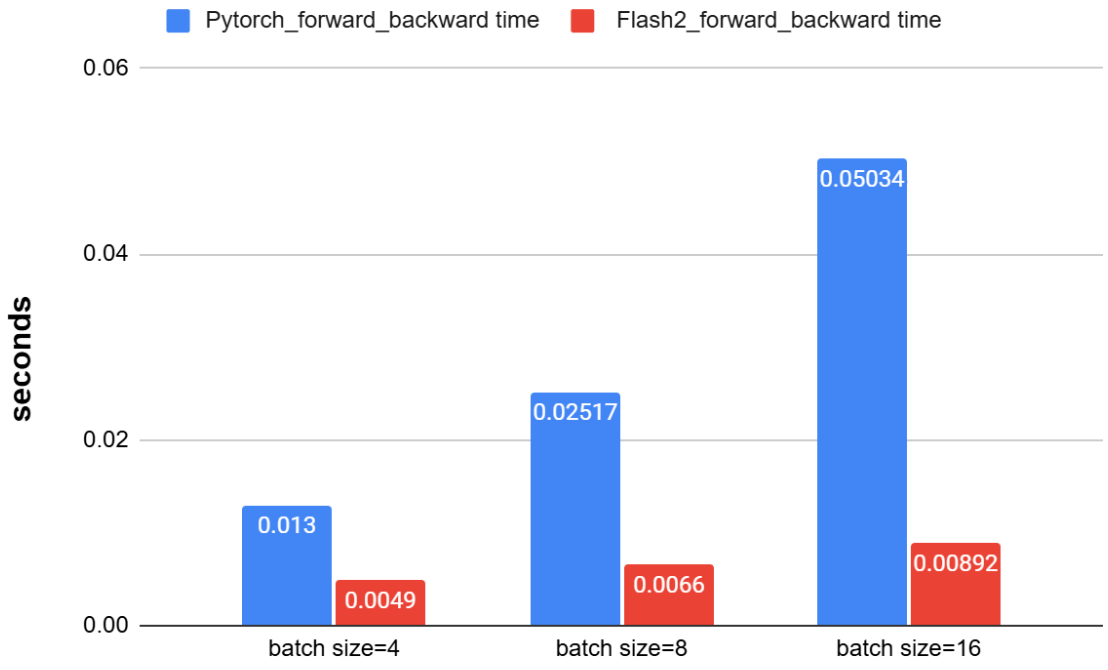
- 透過減少中間張量的生成與存儲，FlashAttention 針對高分辨率或高批次量的訓練場景能顯著降低記憶體壓力。

---

### 3. 調整參數的實驗分析

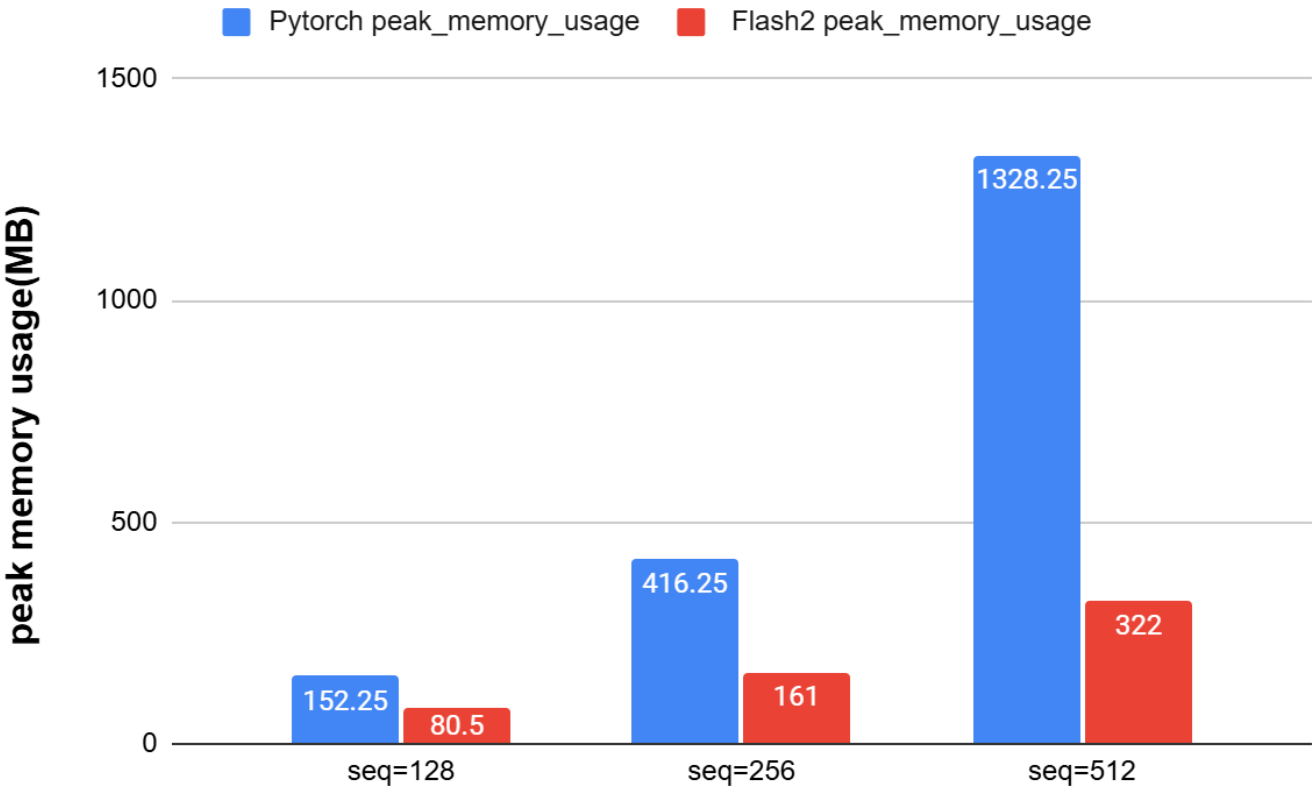
批次大小（**batch_size**）

- 隨著 batch_size 增加，兩種 implementation 的運行時間和 FLOPS 都相應提高。
- 當 batch_size 越來越大，Flash2 提升的 Flops 幅度更大，執行時間略微提升，記憶體用量略微提升。反觀 Pytorch 反而是 Flops 略微提升，執行時間提升幅度較大，記憶體用量大幅增加。可見 Flash2 的計算效能較佳。
-
- 

序列長度（**seq_len**）

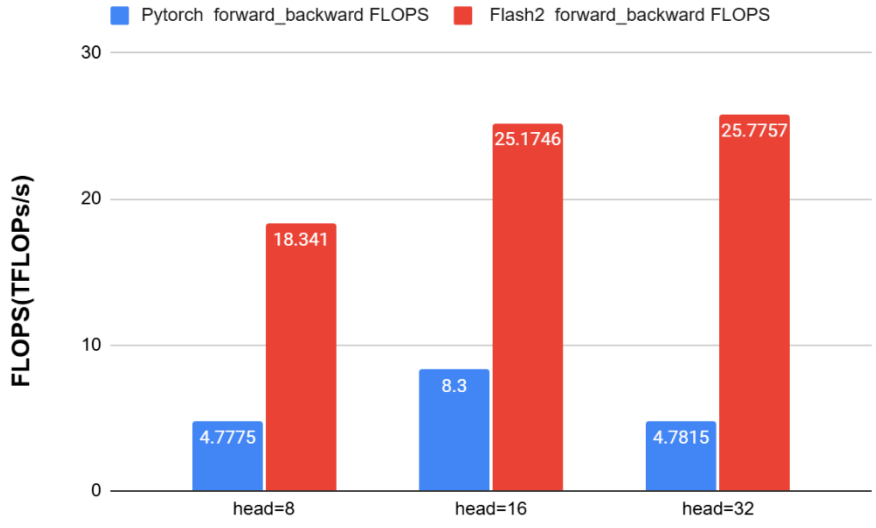
- 當 seq_len 增加，FlashAttention 相較於 PyTorch 更能維持低記憶體使用。反觀 Pytorch 記憶體用量大幅增加，且運行時間也是成倍增加。
- 當 seq_len 增加兩者的 Flops 均沒有顯著增加




- 

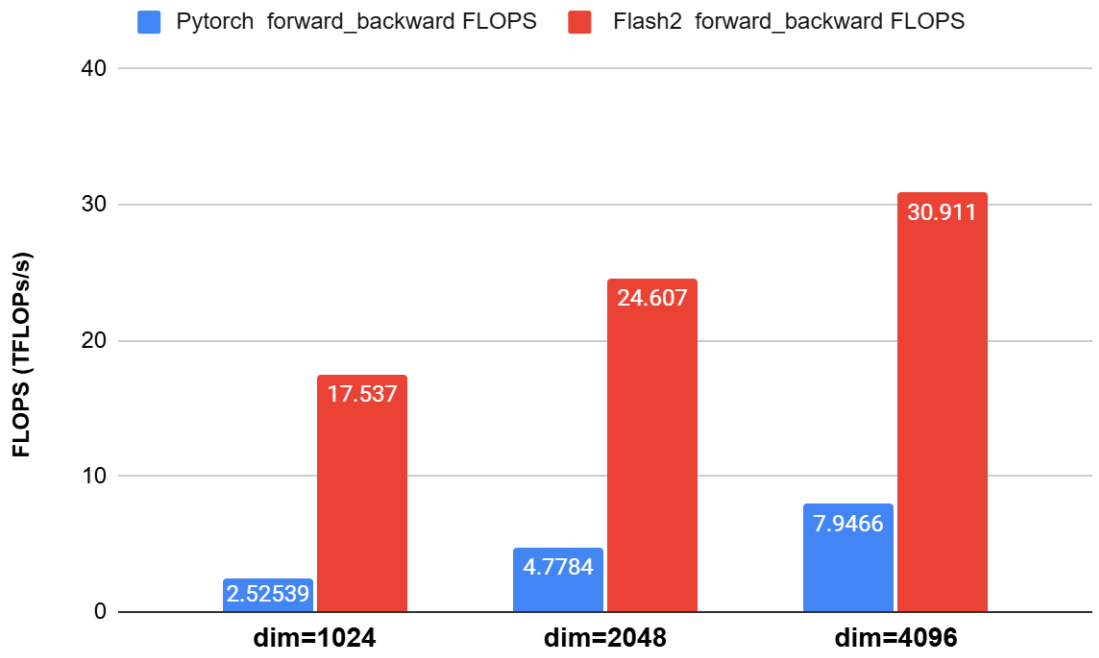
頭部數量（**num_heads**）

- 當 **num_heads** 增加，FlashAttention 相較於 PyTorch 更能維持低記憶體使用。反觀 Pytorch 記憶體用量大幅增加
- 當 num_head 提升時，兩種 implementation，Flops 沒有顯著提升，Pytorch 甚至在 head=32 時 Flop 反而下降了，推測是因為記憶體佔太滿所致。



嵌入維度（**emb_dim**）

- 當 **emb_dim** 增加，FlashAttention 相較於 PyTorch 更能維持低記憶體使用。反觀 Pytorch 記憶體用量大幅增加
- 兩種 implementation 在 **emb_dim** 增加時 Flops 均有提升，而仍是 Flash2 提升幅度較大




---

**4.** 結論

1. **FlashAttention 明顯優於 PyTorch 在運行時間、FLOPS 和記憶體使用上的性能表現**，特別是在大批次、長序列、大嵌入維度的情況下。
2. **對於記憶體敏感型應用**（如超大模型訓練），推薦使用 FlashAttention，以便充分利用硬體資源並降低內存壓力。
3. **在高性能計算環境中，調整批次大小和序列長度的設置能進一步發揮 FlashAttention 的優勢**，特別是在 GPU 訓練環境中可獲得顯著效益。