Saad Ahmad

LLM Inference Optimization - A Case Study Using DistilBERT

Supervisor: Sadegh Keshtkar

Table of Contents

Why Optimize LLM Inference?

- Large models **consume excessive memory** and **have high latency**
- Real-world deployment on **limited hardware** requires optimization
- Key objectives: **reduce latency, memory**, and **maintain accuracy**

Why DistilBERT Instead of LLaMA 2?

- LLaMA 2 lacks readily available classification checkpoints
- Inference-focused project — not fine-tuning
- DistilBERT has pretrained variants on standard tasks
- Ensures **consistent dataset + evaluation** across setups

Hardware and Dataset

- GPU: **NVIDIA A100 (40GB)** on GWDG cluster
- Dataset: **AG News** (text classification)
- Metrics: Accuracy, Latency, Throughput, VRAM
- Sample Size: 10 batches (8 samples each)

Baseline Model

Model: distilbert-base-uncased-finetuned-sst-2-english

- FP32 inference using PyTorch
- Entire batch moved to GPU for evaluation

Quantization: BitsAndBytes (4-bit)

- Applied **post-training quantization**
- Used `bnb_4bit` with `nf4` and `double quant`
- Significantly reduced VRAM usage while slightly improving accuracy
- Regularization effect from quantization likely contributed to performance

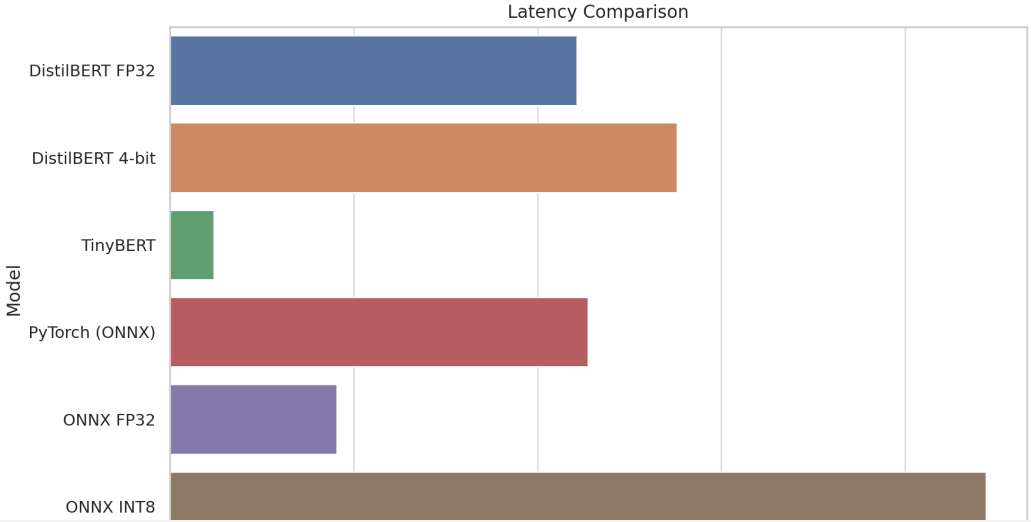
Distilled Model: TinyBERT

- **Smaller than DistilBERT** (4x fewer parameters)
- Extremely fast inference
- Accuracy tradeoff acceptable for edge and low-latency scenarios

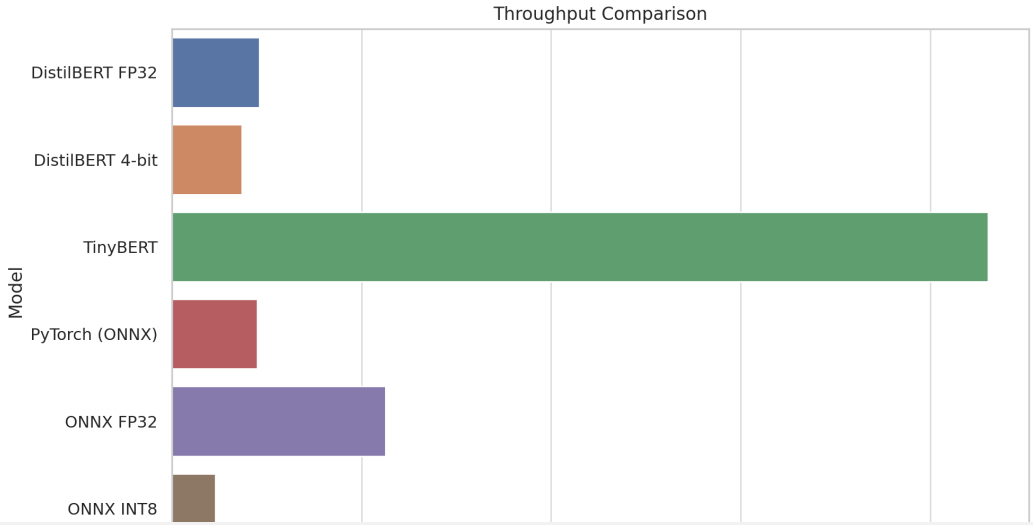
ONNX Runtime (FP32 and INT8)

- DistilBERT exported to ONNX format
- Inference executed using ONNX Runtime backend
- Also evaluated INT8 quantized version (dynamic quantization)

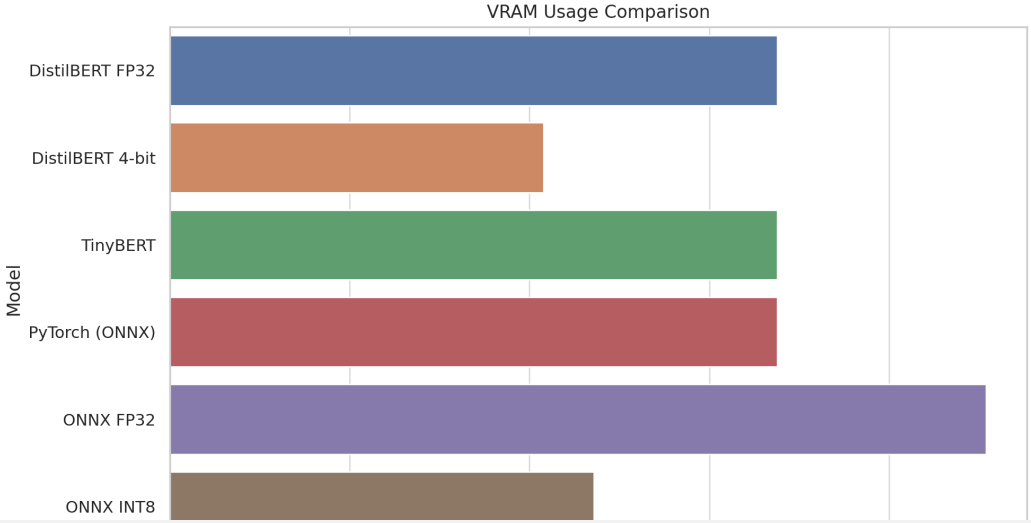
Latency Comparison (↓ is better)



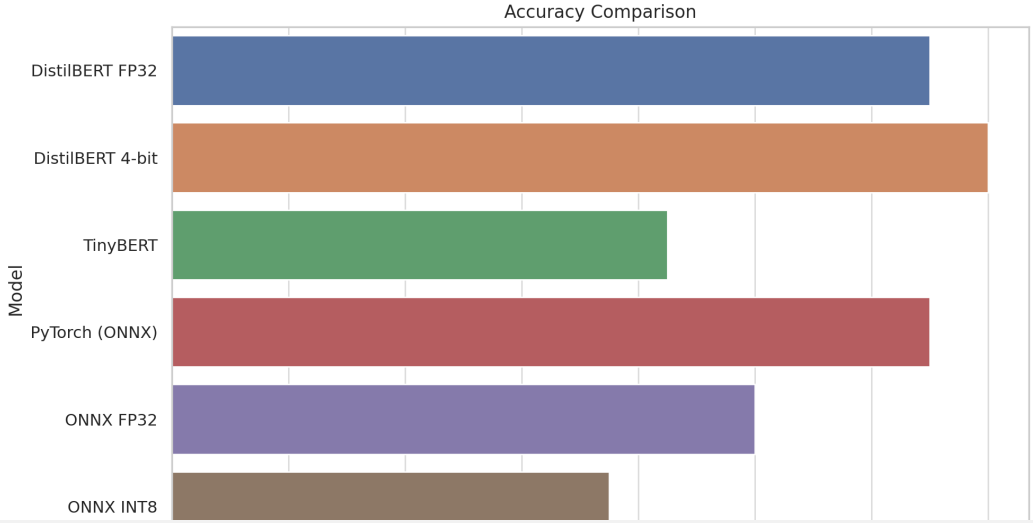
Throughput Comparison (↑ is better)



VRAM Comparison (↓ is better)



Accuracy Comparison (↑ is better)



Performance Results (Tabular Summary)

Model	Latency (s)	Throughput (samples/s)	VRAM (GB)	Acc. (%)
DistilBERT FP32	0.044	23135	1.69	32.5
DistilBERT 4-bit	0.055	18545	1.04	35.0
TinyBERT	0.0048	215234	1.69	21.3
ONNX FP32	0.018	56412	2.27	25.0
ONNX INT8	0.089	11535	1.18	18.8

Observations

- **4-bit BnB** reduced VRAM and slightly improved accuracy
- **TinyBERT** was the fastest but lowest in accuracy
- **ONNX FP32** significantly improved speed with decent accuracy
- **ONNX INT8** reduced memory, but at a high accuracy cost

Key Takeaways

- Optimization involves tradeoffs between speed, memory, and accuracy
- BnB is useful for memory-constrained devices
- ONNX shows consistent speedup over PyTorch
- Tiny models enable real-time inference but compromise performance

Future Work

- Explore serving frameworks (vLLM, TGI)
- Try other quantization techniques (GPTQ, AQLM)
- Benchmark distilled LLaMA family models
- Use larger and diverse evaluation datasets

Conclusion

- Inference optimization is vital for deploying LLMs in production
- DistilBERT provided a robust, consistent benchmarking base
- Tradeoffs revealed by this study help guide deployment choices
- Next steps: expand evaluation scope and try efficient serving stacks