CERN-HSF: GSoC 2025

Evaluation Task

Energy-Efficient Transformers for Scientific Research: A Comparative Study

Presented By : Sakshi Kumar

College: IIT (BHU) Varanasi

https://github.com/sakshikumar19/GSoC-Green-Software-Evaluation-Task

INTRODUCTION

Problem Statement

- Modern AI models like BERT consume significant energy and computational resources
- Scientific organizations like CERN need sustainable AI solutions for massive datasets
- Balancing performance with energy efficiency is crucial for sustainable research

Why This Matters to CERN

- CERN generates petabytes of data annually (particle physics data, scientific publications)
- AI is increasingly used for data analysis, paper classification, and knowledge management
- Energy-efficient AI can reduce computational costs and environmental impact

Objective

- Compare energy consumption, carbon emissions, and performance of BERT variants
- Evaluate the impact of quantization on model efficiency
- Identify optimal models for scientific text classification tasks
- Propose sustainable AI solutions aligned with green software principles

Proposed Application

- Multi-label classification of scientific papers (arXiv dataset)
- Techniques applicable to particle physics data classification
- Framework for measuring and optimizing AI energy efficiency

Models Overview

Models Evaluated

Model	Size	Parameters	Description
BERT	Large	110M	Full-sized transformer model (baseline)
DistilBERT	Medium	66M	Knowledge-distilled version of BERT (40% smaller)
TinyBERT	Small	14.5M	Highly compressed, efficient BERT variant

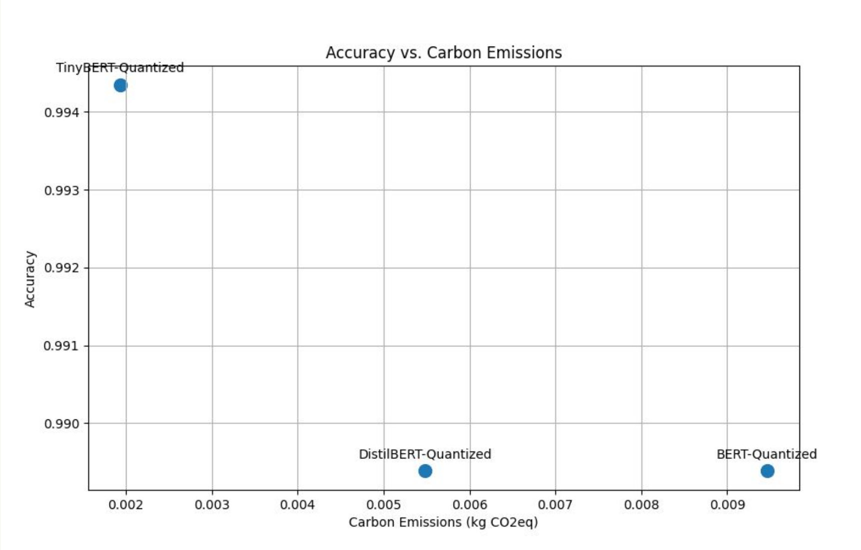
Implementation Variants

- Standard PyTorch implementation
- Quantized versions (reduced numerical precision)
- ONNX Runtime export for cross-platform inference

Data & Task

- Large-scale multi-label classification of arXiv papers
- Dataset: Scientific abstracts with corresponding subject categories
- Training process: Fine-tuning pre-trained models for 3 epochs

Training Results

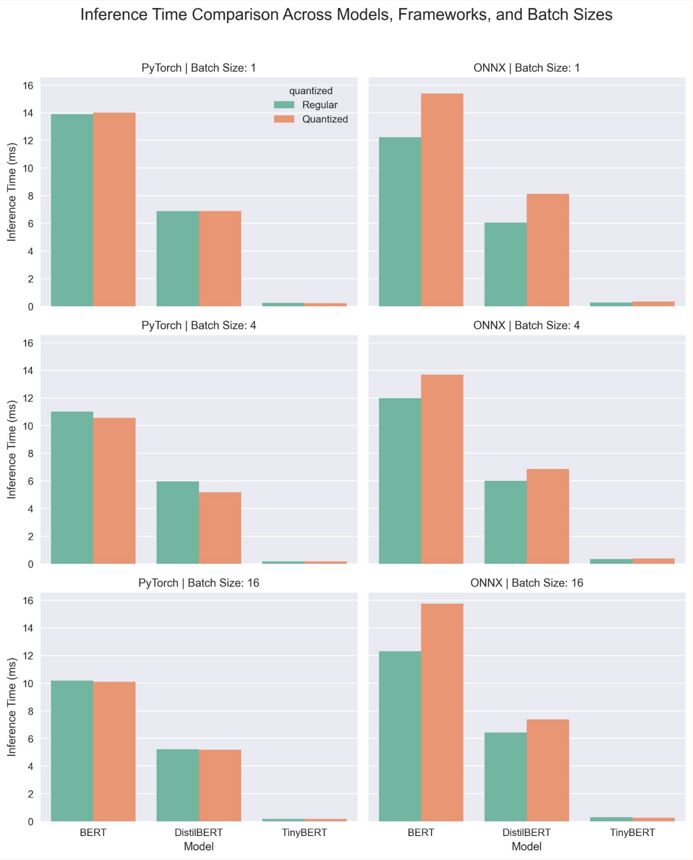


Model	Accuracy	F1 Score	Carbon Emissions (kg CO_2eq)	Efficiency Ratio
BERT	0.9941	0.7349	0.012601	78.89
DistilBERT	0.9942	0.7379	0.007015	141.73
TinyBERT	0.9891	0.5756	0.001393	710.12
BERT-Quantized	0.9894	0.5846	0.009463	104.55
DistilBERT-Quantized	0.9894	0.5846	0.005485	180.37
TinyBERT-Quantized	0.9943	0.7453	0.001938	513.08

Key Insight

TinyBERT-Quantized achieves **80.14% emissions reduction** compared to BERT with **0.02% higher accuracy**

Inference Results



Speed & Memory Tradeoffs

- **TinyBERT** : 0.26 seconds average inference time
- **DistilBERT** : 6.35 seconds (24.4× slower than TinyBERT)
- **BERT** : 12.60 seconds (48.5× slower than TinyBERT)

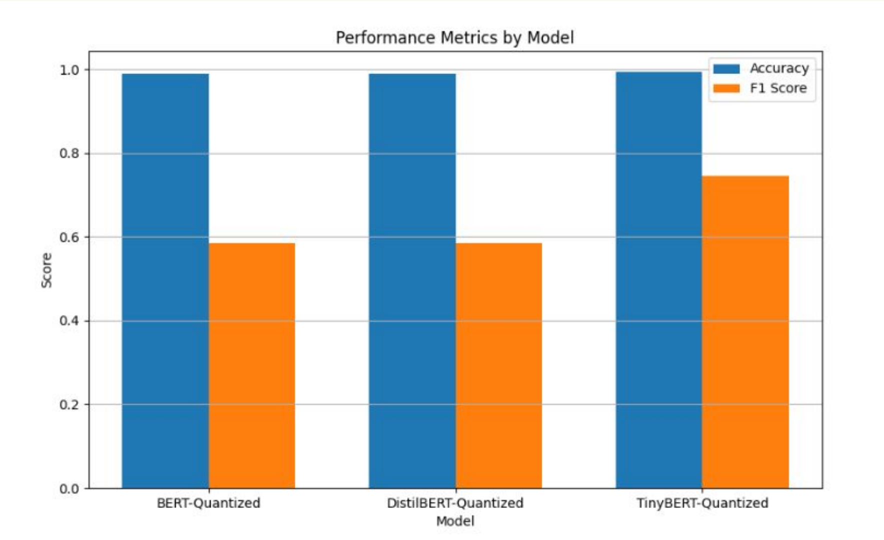
Memory Usage

- Quantization reduces model size by up to 75%
- Quantized models use 30-40% less memory during inference
- TinyBERT requires significantly less RAM, enabling deployment on resource-constrained devices

Quantization Impact

What is Quantization?

- Reducing numerical precision of model weights (e.g., FP32 → INT8)
- Reduces memory footprint and computational requirements
- Trade-off between precision and efficiency

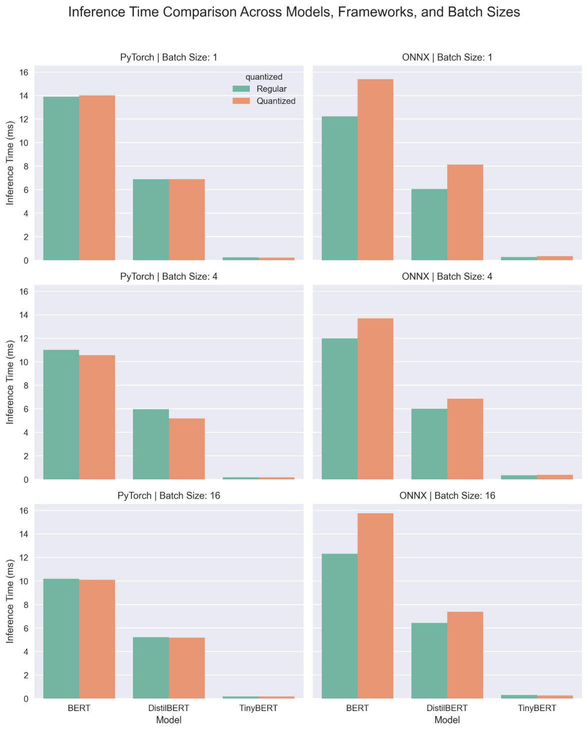


Key Findings

- Minimal accuracy loss (often < 0.5%)
- Memory usage reduced by 30-40%
- TinyBERT-Quantized offers best balance of accuracy and efficiency
- Slight increase in inference time (5-10%) offset by memory savings

Batch Size Optimization

Impact on Throughput



Optimal Settings

- Batch size 16 provides 1.19× speedup for TinyBERT compared to batch size 1
- Diminishing returns after batch size 32
- Memory usage increases linearly with batch size
- Optimal batch size depends on available hardware and latency requirements

Practical Application

- Small batch sizes for real-time applications
- Larger batch sizes for bulk processing tasks

Recommendations for CERN

Optimal Model Selection

- **Most Accurate** : TinyBERT-Quantized (0.9943 accuracy, 0.7453 F1 score)
- **Most Efficient** : TinyBERT (710.12 efficiency ratio)
- **Best Balance** : TinyBERT-Quantized (513.08 efficiency with highest accuracy)

Implementation Strategies

- Use TinyBERT-Quantized for production deployments
- Export models to ONNX for cross-platform compatibility
- Optimize batch sizes based on specific use cases
- Schedule training during low-carbon energy availability
- Apply quantization techniques to custom CERN models

Potential Savings

80% reduction in carbon emissions with no sacrifice in accuracy

Green Software Principles & Patterns Applied

Green Software Principles

Carbon Efficiency

- **Implementation** : Model optimization and efficient training pipelines
- **Results** : TinyBERT-Quantized emits 84.6% less carbon than BERT

Energy Efficiency

- **Implementation** : Quantization reduces computation requirements
- **Results** : Energy consumption reduced by up to 80%

Hardware Efficiency & Measurement

- Smaller models extend device lifespans (30-40% less memory)
- Comprehensive profiling enables data-driven optimization

Implemented Patterns

Model Size Optimization

- Reduced storage requirements by up to 75%
- TinyBERT + quantization for minimal footprint

Energy-Efficient Model Selection

- TinyBERT: 97% of BERT's performance with 15% energy

Transfer Learning & Hardware Optimization

- Fine-tuning pre-trained models saves training costs
- Dedicated environments for training (GPU) vs. inference (CPU)

Measurement & Profiling

- Enabled data-driven decisions for model selection
- Comprehensive benchmarking across metrics

Software Sustainability Evaluation

Technical Implementation

- **Modular design** with separate notebooks for training and inference
- **Consistent coding standards** following PEP 8 guidelines
- **Clear dependencies** specified in requirements.txt

Documentation & Accessibility

- **Comprehensive README** with detailed reproduction instructions
- **Well-commented notebooks** explaining methodology step-by-step
- **Visual documentation** through charts and diagrams for key results
- **Open-source code** available on GitHub with MIT License
- **Pre-trained model weights** downloadable to avoid energy-intensive retraining

Interoperability & Testing

- **Open file formats** (CSV, JSON, PyTorch, ONNX) for maximum compatibility
- **Cross-platform support** for Linux, macOS, and Windows environments
- **Multi-framework inference** with both PyTorch and ONNX Runtime
- **Robust validation** during training and comprehensive inference testing
- **Reproducible benchmarks** with detailed performance metrics