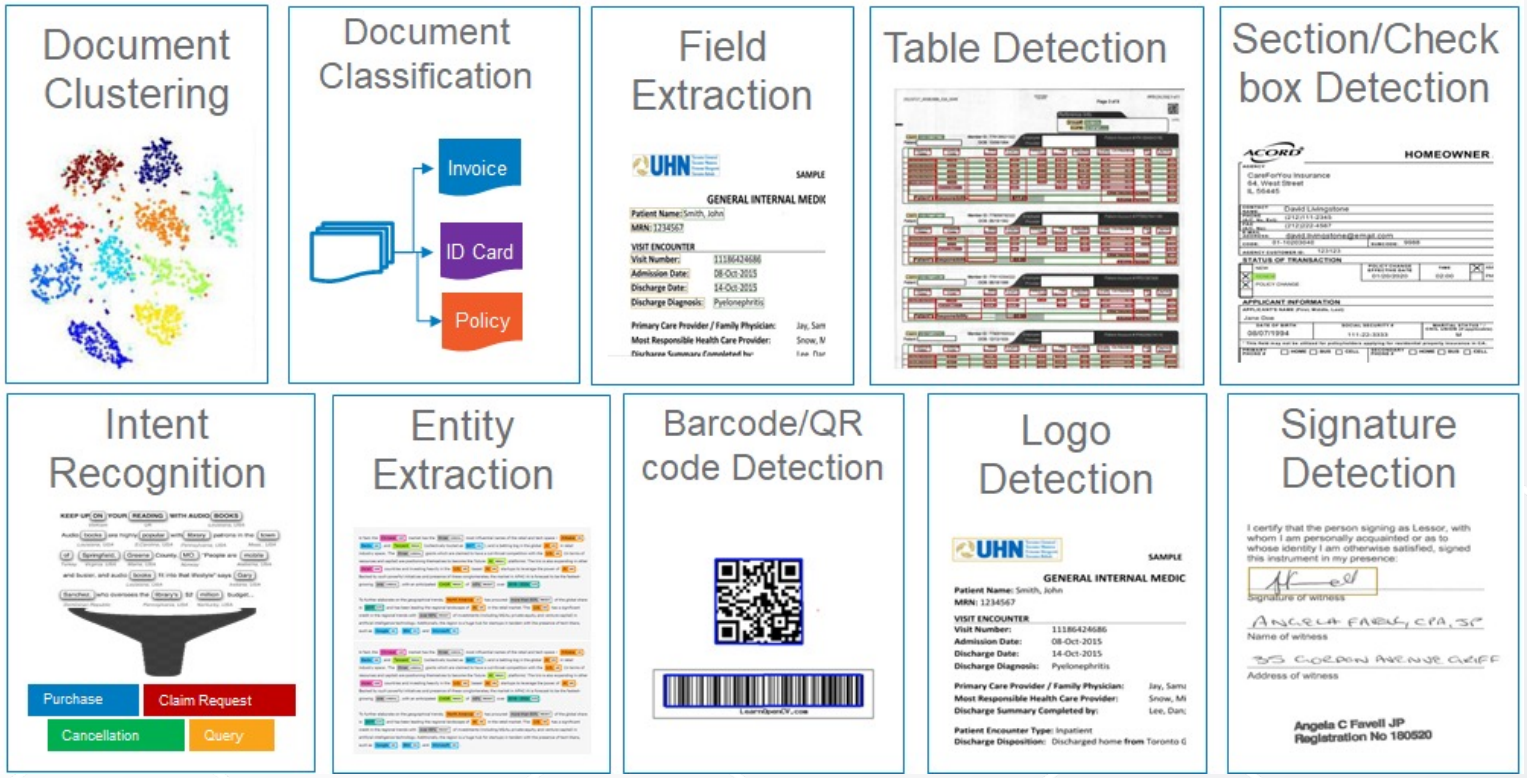
Controlling TCO

MLM Fine tuning & optimisations with oneAPI and 4th Gen Intel® Xeon

Sathish Kumar E V / EdgeVerve

Understanding the landscape



Classification (1) – 200 ms (per document page)
Field Extraction (1) – 150 ms (per document page)
Table detection (1) – 150 ms
Checkbox detection (1) – 150 ms
Intent recognition (1) – 250 ms (100 sentences/page)
Entity recognition (5 entities) – 500ms (100 sentences/page)
16 vCPU, 32 GB

Current state (without optimizations)

- Average 10 AI models per document-page
- Per document-page around 1.4 seconds of AI inferencing
- A 20-page document takes 28 Sec of AI Inferencing
- A batch of 10000 documents (20 pages/doc) takes 78 Hrs of AI inferencing

Objectives

- Background: Around 80% of AI program fail in production. One of the reasons – unviability of the entire solution.

- Approach: Look at the entire DL solutions in two parts – training and inferencing. Training happens intermittent while Inferencing happens continuously.

- Objective: Reduce TCO by
  1. Reduce training cost – by optimizing on CPU than GPU
  2. Reduce inferencing cost - by optimizing Inferencing stack

1. Optimize Training

Fine tuning MLM models

- **Base model: Roberta base**
- **Fine tune for Financial Domain**
- **Dataset: ~ 1.1 M sentences**
- **Training Set: 900384 (80%)**
- **Validation Set: 220872 (20%)**
- **CPU Cores: 224 (across 2 sockets)**
- **Memory: 1 TB**

As – is run on Intel Sapphire Rapids (SPR)

Batch Size	ETA	Max memory usage	Max CPU usage
4	~250 hours	~19.6 GB	~9000% (i.e., 90 CPUs)
8	~160 hours	Not captured	Not captured
16	~180 hours	~35 GB	~10-11K%
256	~ 440 hours	~250 GB	~10-11K%
128	~410 hours	~170 GB	~10-11K%

Based on the above experiments, we realized that despite of sufficient resources (RAM and CPUs) available at disposal, somehow the timings weren't improving.

Fine tuning – Run 2 with optimizations

Training optimizations: IPEX and bfloat16 with auto mixed precision computations.

Batch Size	ETA	Max Mem usage	Max CPU usage
128	~210 hours	~96 GB	~10-11K%
32	~80 hours	~40 GB	~10-11K%

Observations:
50% reduction in training time. Memory consumption significantly reduced.

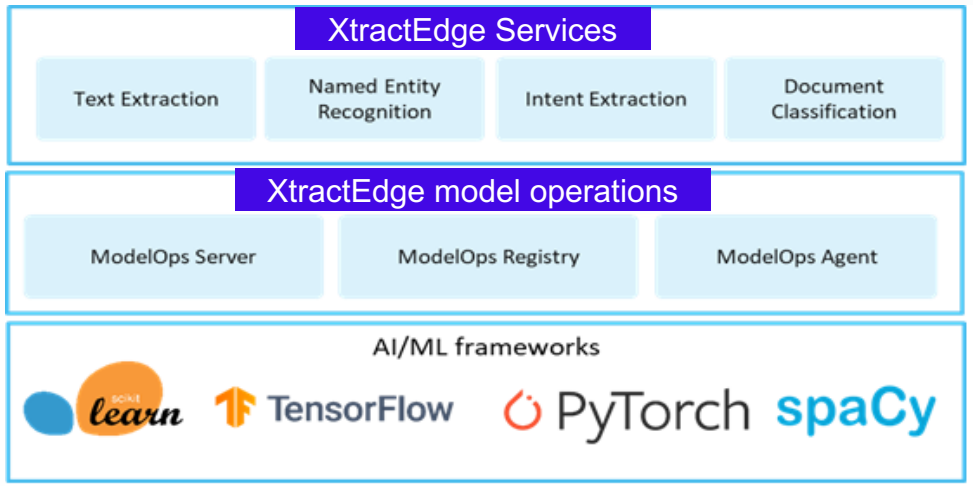
Inference with optimizations: IPEX. Bfloat16 and oneDNN.

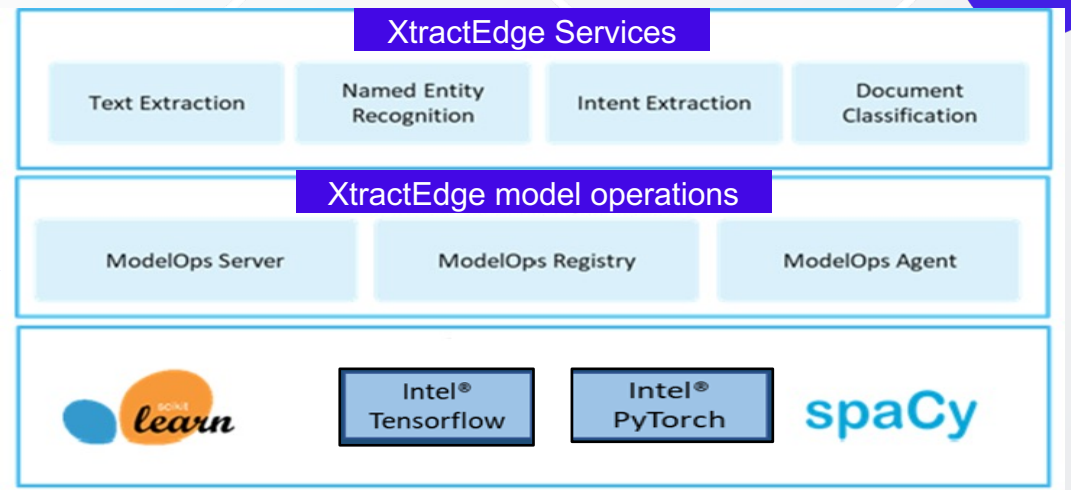
Model	No optimization	With optimization
Time(Batch Size 1)	1.94 s	1.54 seconds
Time(Batch Size 32)	24.64 s	6.88 seconds
TIME Batch Size 64	55.03 s	14.79 seconds
TIME Batch Size 96	80.09 s	26.04 seconds

Observations:
3x improvement on larger batch size.

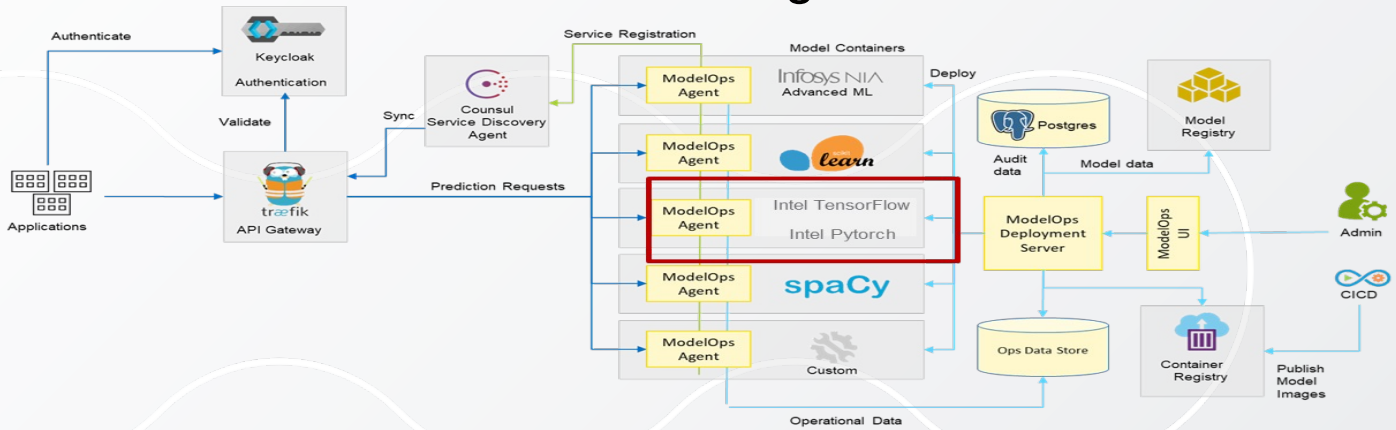
2. Optimize Inferencing

Model operations optimized with Intel



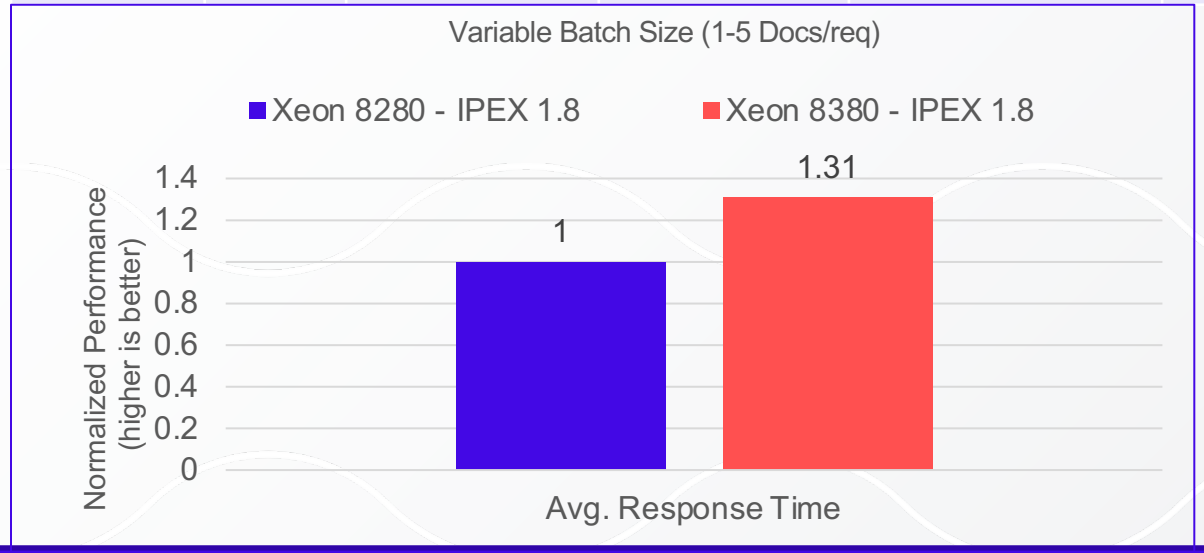


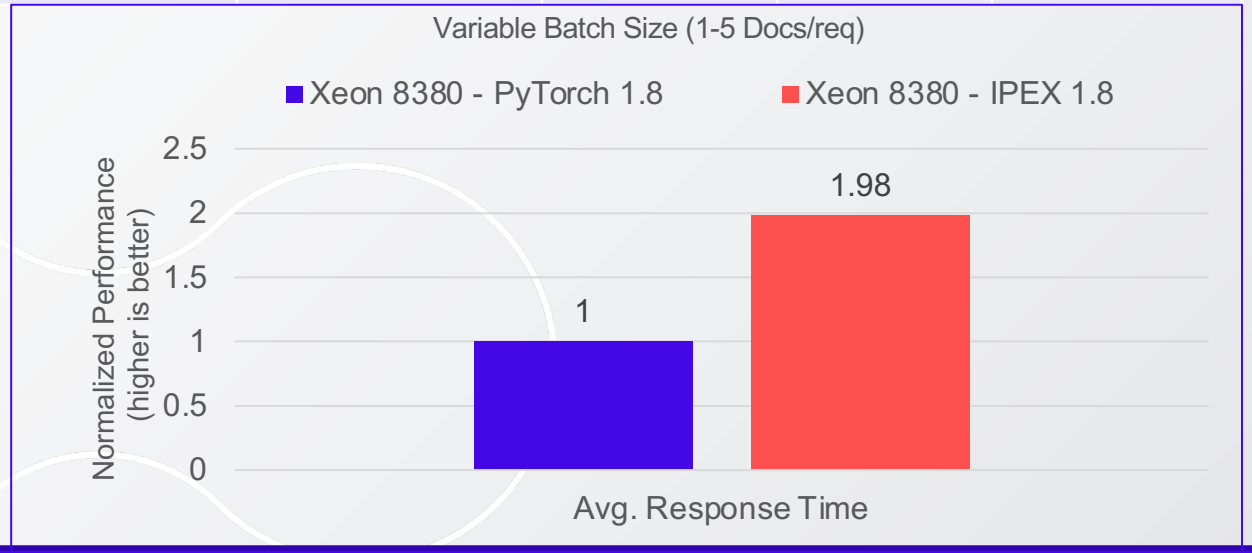
Architecture for serving AI models



Results for field extraction

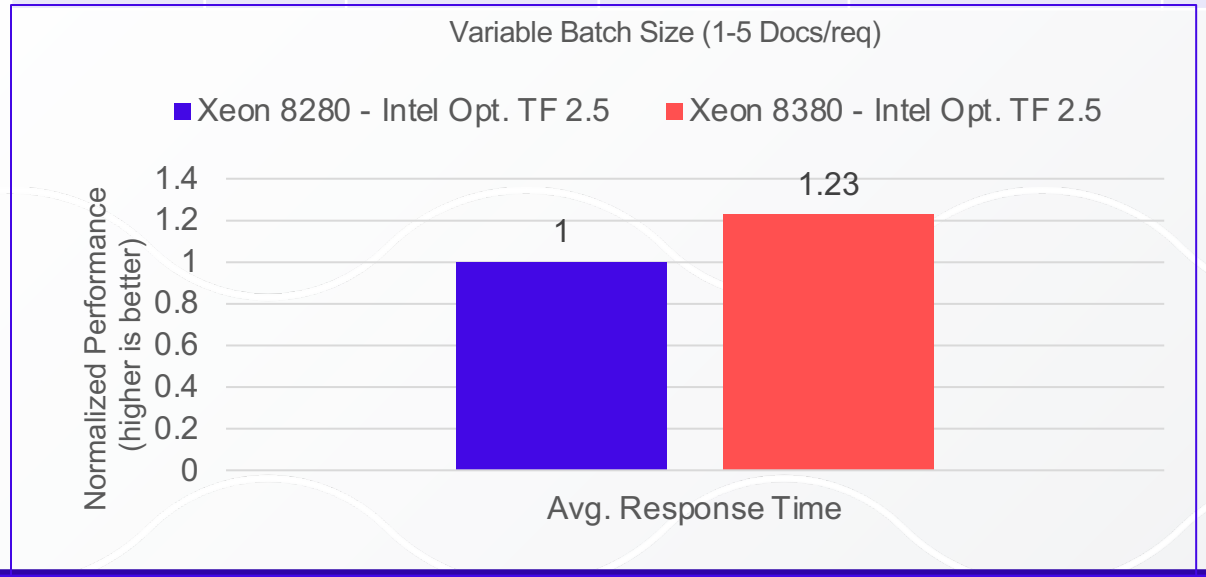
Test case			CONFIG 1 (CLX) Torch Server 1.8	CONFIG 1 (CLX) Intel Python 3.8.5 + Torch Server 1.8 with IPEX 1.8	CONFIG 2 (ICX) Torch Serve 1.8	CONFIG 2 (ICX) Intel Python 3.8.5 + Torch Server 1.8 with IPEX 1.8	Performance Gain (CLX)	Performance Gain (ICX)	Performance Gain (IPEX)
Model	Dataset	Batch size	Avg Response time (ms) / 4K requests (3 docs avg per request)	Avg Response time (ms) / 4K request (3 docs avg per request)	Avg Response time (ms) / 4K request (3 docs avg per request)	Avg Response time (ms) / 4K request (3 docs avg per request)	Native vs IPEX	Native vs IPEX	CLX vs ICX
LayoutLM	Internal	Variable (1-5 documents/req)	527	285	429	217	1.85x	**1.98x**	1.31x

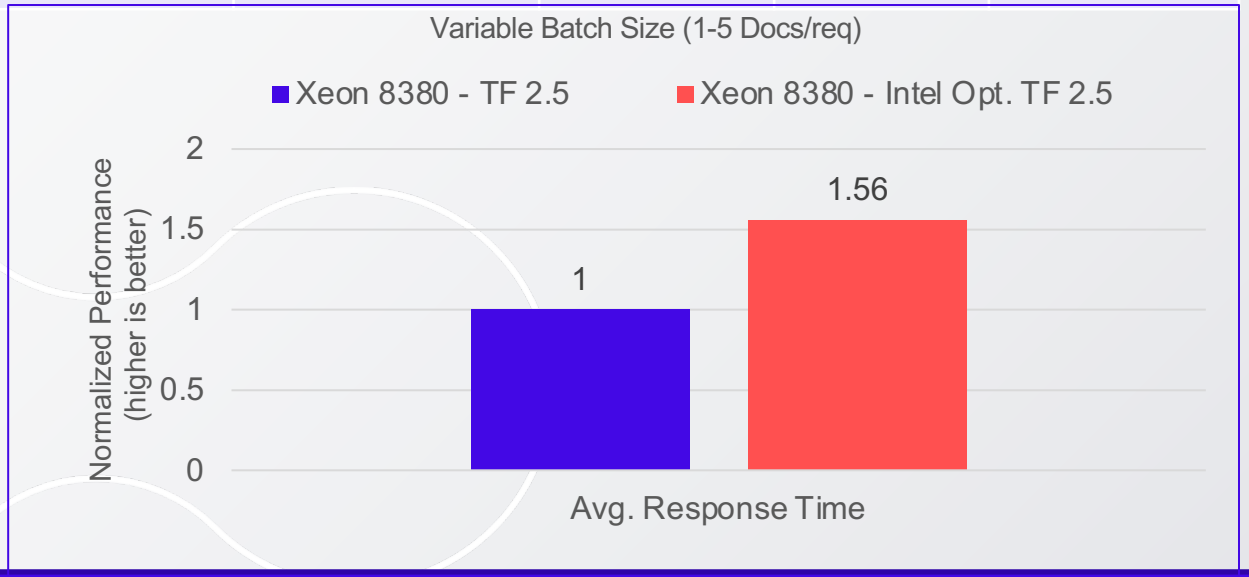




Results for table detection

Test case			CONFIG 1 (CLX) TF Serving 2.5	CONFIG 1 (CLX) Intel Python 3.8.5 + TF 2.5 with Optimization	CONFIG 2 (ICX) TF Serving 2.5	CONFIG 2 (ICX) Intel Python 3.8.5 + TF 2.5 with Optimization	Performance Gain (CLX)	Performance Gain (ICX)	Performance Gain (Intel Opt. TF 2.5)
Model	Dataset	Batch size	Avg Response time (ms) / 4K request (3 docs avg per request)	Avg Response time (ms) / 4K request (3 docs avg per request)	Avg Response time (ms) / 4K request (3 docs avg per request)	Avg Response time (ms) / 4K request (3 docs avg per request)	TF2.5 vs Intel Opt. TF 2.5	TF2.5 vs Intel Opt. TF 2.5	CLX vs ICX
Yolo	Internal	Variable (1-5 documents/req)	464	310	380	254	1.6x	**1.56x**	1.23x





Key Takeaway

- General Purpose CPU's gives us unique advantage to overcome High GPU Cost / Small GPU memory capacity limits to scale out very efficiently & achieve acceptable DL/ML throughput
- With Intel® AI Analytics toolkit optimized for Xeon that covers the full AI pipeline and data science journey from data to training to inference, Many of these multi-fold performance benefits are just 1 line of code change away


oneAPI
Thank you
oneAPI DevSummit for AI & HPC 2023, South Korea