

TENSORFLOW VS PYTORCH: A COMPARATIVE ANALYSIS

GROUP MEMBERS: MORATHI MNKANDLA, ALI SHAN
AHAD

CLASS: DEEP LEARNING ITAI 2376

KEY POINTS:

- TensorFlow (2015) – Developed by Google Brain for scalable, production-ready machine learning.
- PyTorch (2016) – Created by Meta AI with a research-first, dynamic computation approach.
- These two frameworks dominate the AI landscape, driving innovation in both academia and industry.
- TensorFlow focuses on deployment and performance at scale, while PyTorch leads in research flexibility.



ADOPTION IN RESEARCH AND INDUSTRY

Key Stats

- PyTorch powers nearly 80% of NeurIPS research papers that specify a framework (PapersWithCode, 2023).
- TensorFlow remains the industry leader, used by 8.4% of developers (vs 7.9% for PyTorch) in the 2023 Stack Overflow survey.
- The choice of framework often depends on the **goal**:
 - *PyTorch → Rapid experimentation & research.*
 - *TensorFlow → Production pipelines & deployment.*



Slide 04

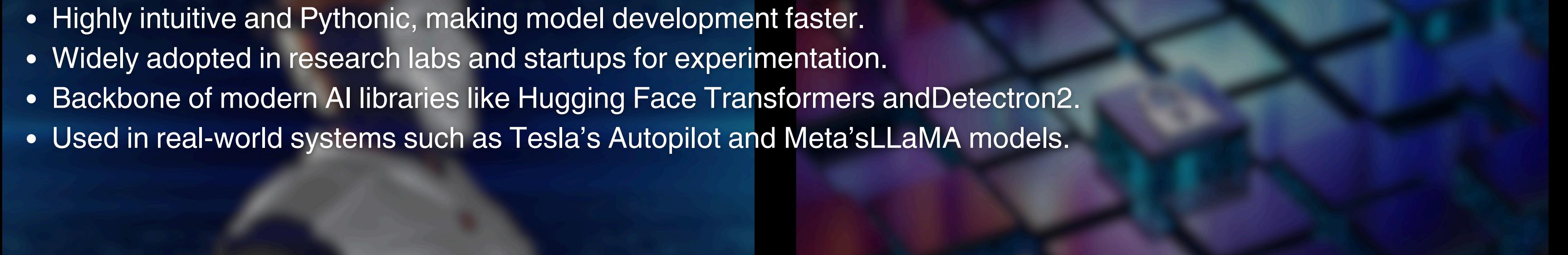
TENSORFLOW OVERVIEW

Points:

- Developed by Google Brain and released in 2015.
- Originated from Google's internal DistBelief project.
- Initially used static computation graphs for performance optimization.
- TensorFlow 2.x (2019) introduced eager execution and integrated Keras for easier model building.
- Deployed across Google products: Search, Gmail, Translate, YouTube, Photos.
- Supported by a strong ecosystem including TensorFlow Lite, TensorFlow.js, and TF Serving.
-

PYTORCH OVERVIEW

Points:

- Released by Meta AI (Facebook) in 2016.
 - Derived from the Torch framework, rewritten in Python.
 - Uses define-by-run (dynamic computation graph) for real-time execution.
 - Highly intuitive and Pythonic, making model development faster.
 - Widely adopted in research labs and startups for experimentation.
 - Backbone of modern AI libraries like Hugging Face Transformers and Detectron2.
 - Used in real-world systems such as Tesla's Autopilot and Meta's LLaMA models.
- 

TOTAL TRAINING TIME COMPARISON

Data Insight:

- PyTorch: 16.98 hours
- TensorFlow: 21.95 hours
- (Values from Novac et al., Sensors 2022.)

Interpretation:

- PyTorch completed training ~23% faster than TensorFlow on the same workload.
- Indicates stronger runtime efficiency and lower overhead for dynamic computation graphs.
- TensorFlow performs competitively but shows slightly higher training latency under similar settings.

EPOCH AND STEP TIMING

Data Insight:

- Average Epoch Time:
 - TensorFlow → 1.53 hours
 - PyTorch → 1.18 hours
- Per Training Step:
 - TensorFlow → 0.00112 s
 - PyTorch → 0.00087 s

Interpretation:

- PyTorch achieves faster per-step execution, especially on GPU-optimized workloads.
- TensorFlow's improvements in version 2.x narrow the gap through XLA and graph optimization.

T

.box

```
1 .box{  
2   position: absolute;  
3   top: 50%;  
4   left: 50%;  
5   transform: translate(-50%, -50%);  
6   width: 400px;  
7   padding: 10px;  
8   background: linear-gradient(45deg, transparent, transparent 4px, black 4px, black 8px);  
9   border-radius: 10px;  
10 }  
11 .box h2{  
12   margin: 0 0 30px;  
13   padding: 0;  
14   color: white;  
15   text-align: center;  
16 }  
17 .box h3{  
18   margin: 0 0 10px;  
19   padding: 0;  
20   color: white;  
21   text-align: center;  
22 }  
23 .box .inputBox{  
24   position: relative;  
25 }  
26 .box .inputBox input{  
27   position: absolute;  
28   top: 50%;  
29   left: 50%;  
30   width: 100px;  
31   height: 100px;  
32   border: none;  
33   border-radius: 50%;  
34   background-color: white;  
35   color: black;
```

INFERENCE FOR SMALL IMAGE MODELS (28x28)

Data Insight:

- TensorFlow (Keras): 0.8985 s/image
- PyTorch: 0.3032 s/image

Interpretation:

- PyTorch achieves $\approx 3x$ faster inference on lightweight models.
- Reflects lower overhead and efficient runtime execution for small data inputs.
- TensorFlow compensates with better integration for mobile and embedded deployment.

```
1 .box{  
2   position: absolute;  
3   top: 50%;  
4   left: 50%;  
5   transform: translate(-50%, -50%);  
6 }  
7  
8 .box h2{  
9   margin: 0 0 30px;  
10  padding: 0;  
11  color: #fff;  
12  text-align: center;  
13 }  
14  
15 .box h3{  
16  margin: 0 0 10px;  
17  padding: 0;  
18  color: #fff;  
19  text-align: center;  
20 }  
21  
22 .box .inputBox{  
23  position: relative;  
24 }  
25  
26 .box .inputBox input{  
27  width: 100px;  
28  height: 40px;  
29  border: none;  
30  background-color: #f0f0f0;  
31  border-radius: 10px;  
32  font-size: 16px;  
33  color: #333;  
34  transition: border-color 0.3s;  
35 }  
36  
37 .box .inputBox input::placeholder{  
38  opacity: 0.5;  
39  color: #333;  
40 }  
41  
42 .box .inputBox .error{  
43  position: absolute;  
44  top: 50%;  
45  left: 50%;  
46  transform: translate(-50%, -50%);  
47  background-color: #e0e0e0;  
48  border: 1px solid #ccc;  
49  padding: 5px;  
50  border-radius: 5px;  
51  font-size: 14px;  
52  color: #333;  
53  font-weight: bold;  
54  text-align: center;  
55 }  
56  
57 .box .inputBox .error::before{  
58  content: "✖";  
59  font-size: 18px;  
60  color: #c00;  
61  margin-right: 5px;  
62 }  
63  
64 .box .inputBox .error .text{  
65  margin-top: 5px;  
66  font-size: 12px;  
67  color: #333;  
68 }  
69  
70 .box .inputBox .error .button{  
71  position: absolute;  
72  bottom: 10px;  
73  right: 10px;  
74  background-color: #333;  
75  color: #fff;  
76  border: none;  
77  border-radius: 50%;  
78  width: 20px;  
79  height: 20px;  
80  display: flex;  
81  align-items: center;  
82  justify-content: center;  
83  font-size: 14px;  
84  font-weight: bold;  
85  text-decoration: none;  
86 }  
87  
88 .box .inputBox .error .button::before{  
89  content: "X";  
90  font-size: 12px;  
91  color: #fff;  
92  margin-right: 5px;  
93 }  
94  
95 .box .inputBox .error .button .text{  
96  margin-top: 5px;  
97  font-size: 10px;  
98  color: #333;  
99 }  
100  
101 .box .inputBox .error .button .text .link{  
102  color: #333;  
103  text-decoration: underline;  
104 }  
105  
106 .box .inputBox .error .button .text .link::hover{  
107  color: #333;  
108  text-decoration: underline;  
109 }
```

INFERENCE FOR LARGER MODELS & DEPTH NETWORKS

Data Insight:

- 64x64 images:
 - TensorFlow → 2.0182 s
 - PyTorch → 1.5961 s
- Depth model test:
 - TensorFlow → 2.667 s
 - PyTorch → 1.174 s

Interpretation:

- PyTorch consistently maintains lower inference latency for larger models.
- TensorFlow, however, supports hardware acceleration (TPU/XLA) that can offset raw speed differences.

OVERALL FEATURE COMPARISON (SCALE OF 0-10)

TensorFlow:

- Usability → 7
- Deployment → 9
- Performance → 8
- Community → 8

PyTorch:

- Usability → 9
- Deployment → 7
- Performance → 8
- Community → 9

Summary:

- PyTorch dominates in usability and research community engagement.
- TensorFlow excels in deployment and ecosystem integration.
- Performance is effectively equal when optimized properly.

DEPLOYMENT STRENGTHS (0 = WEAK, 3 = STRONG)

TensorFlow Strengths:

- Mobile / Edge – 3 (via TensorFlow Lite)
- Server – 3 (via TensorFlow Serving)
- Browser – 3 (TensorFlow.js)
- TPU Support – 3 (native integration)
- ONNX Interoperability – 2

PyTorch Strengths:

- Mobile / Edge – 1
- Server – 2
- Browser – 1
- TPU Support – 1 (via XLA backend)
- ONNX Interoperability – 3

Summary:

- TensorFlow dominates in deployment and portability.
- PyTorch leads in cross-framework flexibility with ONNX.

FINAL COMPARATIVE SUMMARY

Highlights:

- PyTorch is ideal for researchers and developers seeking agility and dynamic modeling.
- TensorFlow remains the best for scalable production systems and cross-platform deployment.
- Performance parity exists for well-optimized models.
- Both frameworks continue to evolve — their ecosystems now overlap through ONNX and shared APIs.
- Recommendation:
 - Prototype in PyTorch
 - Deploy in TensorFlow (or via ONNX pipelines)

REFERENCES & RESOURCE

Sources Cited:

- Ba Alawi – Comparative Survey of PyTorch vs TensorFlow (2025).
- Novac et al. (2022) – Sensors Journal: training and inference benchmarks.
- Becirovic et al. (2021) – comparative inference analysis for CNNmodels.
- PapersWithCode (2023) – framework usage in NeurIPS publications.
- Stack Overflow Developer Survey (2023) – AI framework adoption rates.
- TensorFlow & PyTorch official documentation for feature verification.



A hand is shown interacting with a robotic arm against a dark background. The background features several glowing pink energy fields or particles. The hand is positioned near the robotic arm's gripper, which is holding a small object. The overall theme suggests a futuristic or advanced technological interaction.

**THANK YOU
FOR YOUR ATTENTION**