CSCE 585: Machine Learning Systems

Lecture 8: Replicating Results in Machine Learning Systems Research

Pooyan Jamshidi

UNIVERSITY OF South Carolina

The Importance of Replication in MLSys Research

- Why Replicate?
  - Validates and strengthens research findings.
  - Enhances understanding of methodologies.
  - Promotes transparency and reproducibility in science.
- Lecture Overview:
  - Steps to replicate results from an MLSys paper.
  - Using the InferLine paper as a practical example

Introducing InferLine

- **Full Title:** "InferLine: Latency-Aware Provisioning and Scaling for Prediction Serving Pipelines"
- **Authors & Venue:** Crankshaw et al., SoCC 2020
- **Key Contributions:**
  - A system for efficient provisioning and management of ML inference pipelines.
  - Addresses latency constraints and cost efficiency.
- **Relevance:**
  - Combines systems engineering with machine learning.
  - Provides a rich case study for replication efforts.

Core Concepts and Goals

- **Problem Statement:**
  - Managing ML inference pipelines under tight tail latency SLOs.
  - Handling hardware heterogeneity and bursty workloads.
- **InferLine Components:**
  - Low-Frequency Planner:
    - Optimizes pipeline configurations periodically.
  - High-Frequency Tuner:
    - Adapts to workload changes in real-time.
- **Expected Outcomes:**
  - Cost-efficient resource allocation.
  - High latency SLO attainment (>99%).

Methodological Approach

1. **In-Depth Reading:**
   - Understand the research question and methodology.
   - Identify key experiments and results to replicate.
2. **Resource Gathering:**
   - Access code repositories and datasets.
   - Note any dependencies or specific hardware requirements.
3. **Environment Setup:**
   - Replicate the original experimental environment as closely as possible.
   - Install necessary software and configure hardware.
4. **Re-Implementation (if needed):**
   - If code is unavailable, implement methods based on descriptions.

Methodological Approach

5. **Experimentation:**
    - Run the experiments following the original protocols.
    - Collect data systematically.
6. **Analysis and Comparison:**
    - Compare replicated results with those reported.
    - Discuss any discrepancies or deviations.
7. **Documentation:**
    - Keep detailed records of all steps and observations.
    - Prepare to share findings with others.

Setting Up for InferLine Replication

- **Hardware Requirements:**
  - CPUs, GPUs (e.g., NVIDIA Tesla K80), potentially TPUs or FPGAs.
- **Software Tools:**
  - Prediction serving frameworks (Clipper, TensorFlow Serving).
  - Simulation tools (if applicable).
- **Models and Datasets:**
  - Pre-trained models: ResNet152, NMT models, etc.
  - Datasets: ImageNet samples, translation corpora.
- **Configuration Parameters:**
  - Batch sizes, hardware accelerators, replication factors.

Replicating Experiment 1 – Model Profiling

- **Objective:**
  - Measure throughput and latency of models on various hardware.
- **Procedure:**
  - Run inference tasks with different batch sizes on CPU and GPU.
  - Record throughput (QPS) and P99 latency.
- **Expected Observations:**
  - GPUs offer higher throughput with larger batch sizes.
  - Latency may increase with batch size due to processing delays.
- **Data Visualization:**
  - Graphs of throughput vs. batch size.
  - Latency vs. batch size charts.

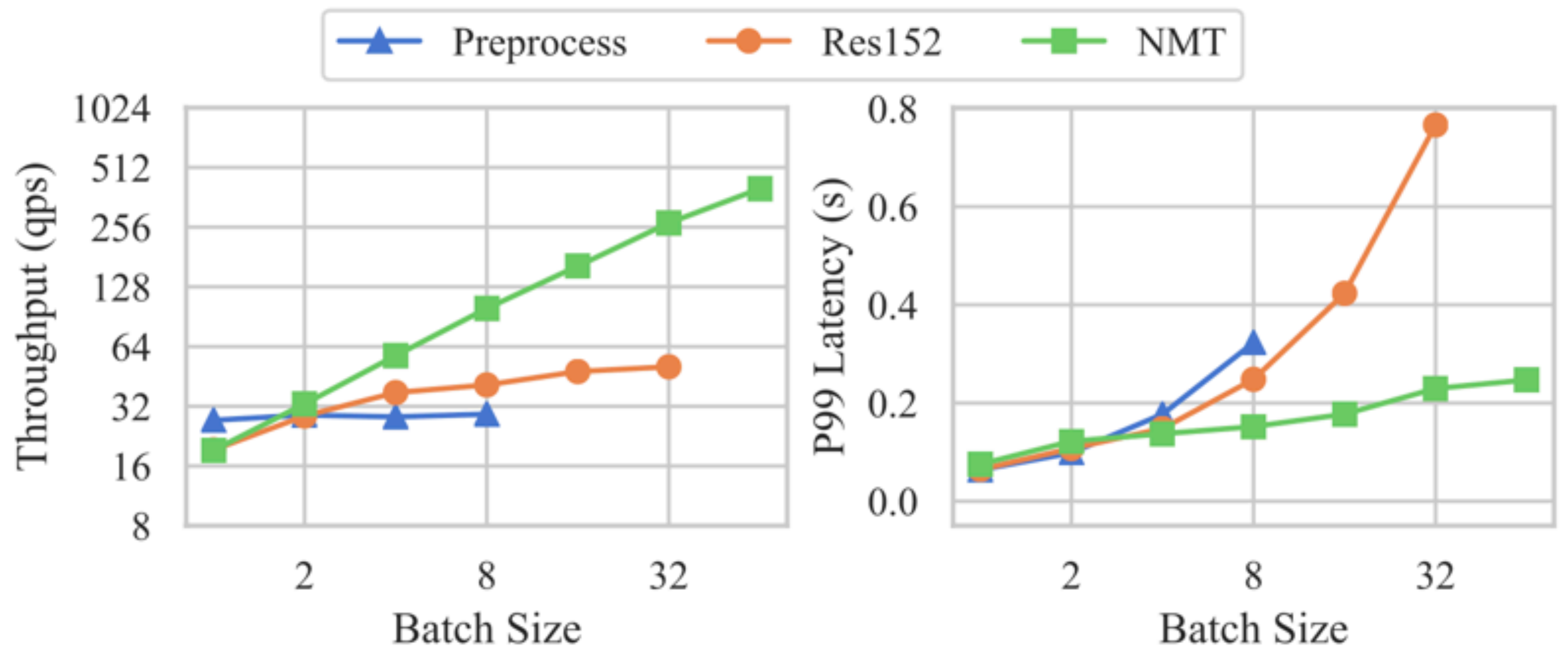
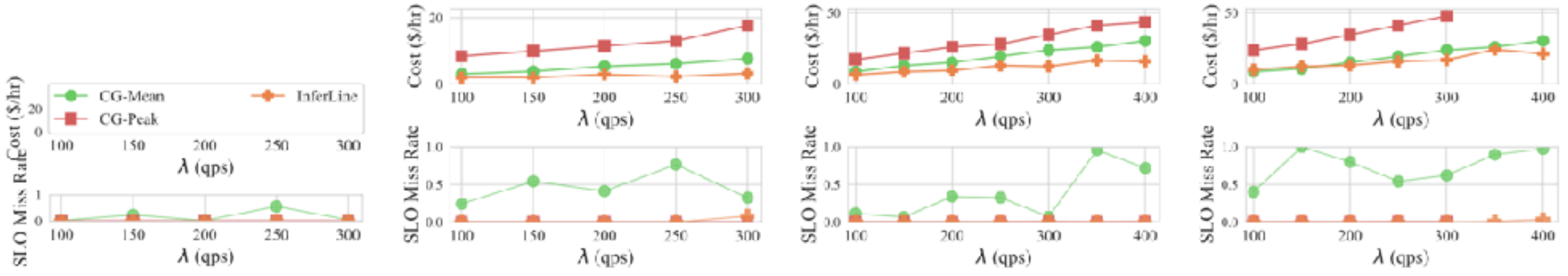


Figure 3. Example Model Profiles on K80 GPU. The preprocess model has no internal parallelism and cannot utilize a GPU. Thus, it sees no benefit from batching. Res152 (image classification) & TF-NMT(text translation model) benefit from batching on a GPU but at the cost of increased latency.

Replicating Experiment 2 – Pipeline Provisioning

- **Objective:**
  - Assess cost and latency performance of different pipeline configurations.
- **Procedure:**
  - Implement InferLine's low-frequency planner.
  - Compare against baseline methods (e.g., static provisioning).
  - Test under varying workloads, including bursty traffic.
- **Metrics:**
  - Cost efficiency (resource utilization vs. expense).
  - Latency SLO attainment (percentage of queries meeting latency targets).
- **Data Visualization:**
  - Cost comparison bar charts.
  - Latency attainment graphs.

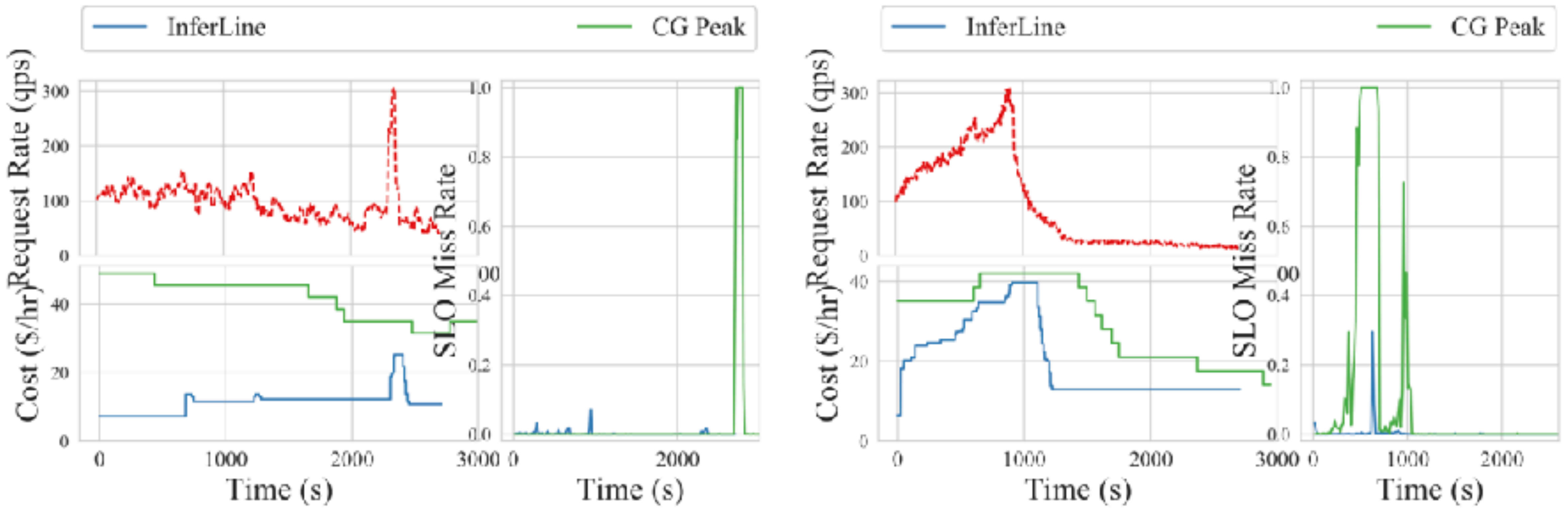


(a) Image Processing CV 1.0 (b) Image Processing CV 4.0 (c) Video Monitoring CV 1.0 (d) Video Monitoring CV 4.0

Figure 5. Comparison of InferLine's Planner to coarse-grained baselines (150ms SLO) InferLine outperforms both baselines, consistently providing both the lowest cost configuration and highest SLO attainment (lowest miss rate). CG-Peak was not evaluated on $\lambda > 300$ because the configurations exceeded cluster capacity.

Replicating Experiment 3 – High-Frequency Tuning

- **Objective:**
  - Demonstrate how the high-frequency tuner adapts to workload spikes.
- **Procedure:**
  - Simulate sudden increases in query rates.
  - Monitor how the tuner adjusts resources in real-time.
- **Expected Results:**
  - Maintenance of latency SLOs during spikes.
  - Efficient resource scaling without over-provisioning.
- **Data Visualization:**
  - Time-series plots showing resource adjustments.
  - Latency over time during workload changes.



(a) AutoScale Big Spike

(b) AutoScale Slowly Varying

Figure 6. Performance comparison of the high-frequency tuning algorithms on traces derived from real workloads. These are the same workloads evaluated in [12] which forms the basis for the coarse-grained baseline. Both workloads were evaluated on the Social Media pipeline with a 150ms SLO. In Fig. 6(a), InferLine maintains a 99.8% SLO attainment overall at a total cost of $8.50, while the coarse-grained baseline has a 93.7% SLO attainment at a cost of $36.30. In Fig. 6(b), InferLine has a 99.3% SLO attainment at a cost of $15.27, while the coarse-grained baseline has a 75.8% SLO attainment at a cost of $24.63, a 34.5x lower SLO miss rate.

Challenges Encountered

- **Hardware Limitations:**
  - Issue: Different hardware from the original setup.
  - Solution: Adjust configurations; note differences in results.
- **Software Dependencies:**
  - Issue: Outdated or unavailable software versions.
  - Solution: Find compatible alternatives; document changes.
- **Incomplete Details:**
  - Issue: Missing parameters or settings in the paper.
  - Solution: Make educated guesses; reach out to authors if possible.

Analyzing and Comparing Results

- **Result Comparison:**
  - Place your findings side-by-side with the original results.
  - Use tables and graphs for clarity.
- **Discrepancy Analysis:**
  - Identify factors contributing to any differences.
  - Discuss the impact of environment and implementation variations.
- **Validation:**
  - Confirm whether key trends and conclusions hold.
  - Reflect on the robustness of the original findings.

Lessons Learned from Replication

- **Technical Skills Enhanced:**
  - Deepened understanding of ML systems and infrastructure.
- **Research Skills Developed:**
  - Critical analysis of methodologies.
  - Problem-solving in face of incomplete information.
- **Importance of Replicability:**
  - Reinforces the need for detailed reporting in research.
  - Encourages openness and data sharing.

Applying the Experience to Your Projects

- **Select a Target Paper:**
    - Choose an MLSys paper relevant to your project interests.
- **Plan Your Replication Strategy:**
    - Identify key experiments to focus on.
    - Outline necessary resources and steps.
- **Collaborate and Seek Guidance:**
    - Work with peers or mentors.
    - Don't hesitate to ask for help or clarification.
- **Document Thoroughly:**
    - Keep detailed records for your report and presentation.

The Value of Replicating MLSys Research

- **Reinforces Learning:**
  - Solidifies understanding of complex systems.
- **Contributes to the Field:**
  - Helps verify and validate existing research.
  - Identifies potential areas for improvement.
- **Promotes Scientific Integrity:**
  - Encourages transparency and accountability.

Detailed Real-World Examples of Replication Leading to Advancements

ImageNet and the Advancement of Deep Learning

Original Work: **AlexNet** by Krizhevsky et al. (2012)

- **Achievement**: Won the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) 2012 with a top-5 error rate significantly lower than previous winners.
- **Key Innovations**:
  - Deep convolutional neural network with eight layers.
  - Use of Rectified Linear Units (ReLU) for activation functions.
  - Implementation of dropout to prevent overfitting.

Replication and Extension:

- **Replication Efforts**:
  - Researchers worldwide replicated AlexNet to understand its architecture and training methodologies.
  - Open-source frameworks like Caffe and TensorFlow included AlexNet implementations, facilitating replication.
- **Advancements from Replication**:
  - **VGGNet** (Simonyan and Zisserman, 2014): Explored the effect of convolutional network depth on accuracy by using very small convolution filters, leading to improved performance.
  - **GoogLeNet/Inception Modules** (Szegedy et al., 2015): Introduced a more efficient architecture using inception modules, which allowed for increased depth and width while keeping computational costs manageable.
  - **ResNet** (He et al., 2015): Addressed the degradation problem in deep networks using residual connections, enabling the training of networks with over 150 layers.

Significance:

- Replication of AlexNet validated the effectiveness of deep learning models and inspired further research into network architectures.
- Led to rapid advancements in computer vision and the widespread adoption of deep learning techniques across various domains.

Transformer Models and Progress in Natural Language Processing

- **Original Work**: **Transformer** architecture by Vaswani et al. (2017)
  - **Achievement**: Introduced a novel architecture based solely on attention mechanisms, dispensing with recurrence and convolutions entirely.
  - **Key Innovations**:
    - Self-attention mechanism to model relationships between all words in a sentence, regardless of their position.
    - Positional encoding to retain the order of the sequence.
- **Replication and Extension**:
  - **Replication Efforts**:
    - Researchers replicated the Transformer model to understand its capabilities in sequence modeling.
    - Implementations were made available in libraries like TensorFlow and PyTorch, aiding replication.
  - **Advancements from Replication**:
    - **BERT** (Devlin et al., 2018): Built upon the Transformer architecture to create bidirectional representations, achieving state-of-the-art results on multiple NLP tasks.
    - **GPT Series** (Radford et al., 2018-2020): Used Transformer decoders to create powerful language models capable of generating coherent text.
    - **Transformer-XL** (Dai et al., 2019): Addressed the fixed-length context limitation, enabling modeling of longer-term dependencies.
    - **Reformer** (Kitaev et al., 2020): Made Transformers more efficient for long sequences using locality-sensitive hashing.
- **Significance**:
  - Replication allowed researchers to explore and understand the strengths and limitations of the Transformer architecture.
  - Led to breakthroughs in machine translation, language understanding, and text generation.

Generative Adversarial Networks (GANs)

- **Original Work**: **GANs** by Goodfellow et al. (2014)
  - **Achievement**: Introduced a framework where two neural networks, a generator and a discriminator, are trained simultaneously to produce realistic data.
  - **Key Innovations**:
    - Adversarial training that pits two networks against each other.
    - Potential to generate data indistinguishable from real data.
- **Replication and Extension**:
  - **Replication Efforts**:
    - Due to the novelty, many researchers replicated GANs to understand their training dynamics.
    - Faced challenges like mode collapse and training instability, leading to improvements.
  - **Advancements from Replication**:
    - **DCGAN** (Radford et al., 2015): Provided architectural guidelines for stable GAN training, enabling the generation of higher-quality images.
    - **Wasserstein GAN** (Arjovsky et al., 2017): Introduced a new loss function to improve training stability.
    - **StyleGAN** (Karras et al., 2019): Enabled control over the style and features of generated images, producing photorealistic human faces.
- **Significance**:
  - Replication led to a deeper understanding of GANs, solving initial training difficulties.
  - Opened up new applications in image synthesis, style transfer, and data augmentation.

Adversarial Examples and Model Robustness

- **Original Work**: **Adversarial Attacks** by Szegedy et al. (2013)
  - **Achievement**: Demonstrated that adding imperceptible perturbations to input data could cause neural networks to make incorrect predictions.
  - **Key Innovations**:
    - Highlighted vulnerabilities in neural networks.
    - Raised concerns about the security and reliability of AI systems.
- **Replication and Extension**:
  - **Replication Efforts**:
    - Researchers replicated adversarial attacks to assess the extent of the vulnerability.
    - Explored different methods to generate adversarial examples.
  - **Advancements from Replication**:
    - **Development of Defense Mechanisms**:
      - **Adversarial Training**: Training models with adversarial examples to improve robustness.
      - **Defensive Distillation**: Using knowledge distillation to reduce sensitivity to perturbations.
    - **Certification Methods**:
      - Providing guarantees about a model's robustness within certain perturbation bounds.
    - **Robust Architecture Design**:
      - Designing network architectures inherently more resistant to adversarial attacks.
- **Significance**:
  - Replication efforts led to a better understanding of the weaknesses in neural networks.
  - Spurred a new research area focused on model robustness and security.

Impact of Replication in These Examples

- **Validation of Results**: Replication confirmed the reliability of original findings, increasing confidence in the methods and conclusions.
- **Identification of Limitations**: Helped uncover shortcomings or conditions under which the original results may not hold, guiding future research directions.
- **Innovation through Extension**: By building upon replicated work, researchers developed new models, techniques, and applications, driving the field forward.
- **Enhanced Collaboration**: Replication efforts often lead to collaborations between original authors and replicators, fostering a more cohesive research community.

Encouraging Replication in Your Work

- **Transparency**: Share code, data, and detailed methodologies to facilitate replication by others.
- **Documentation**: Keep thorough records of experiments, parameters, and environment configurations.
- **Collaboration**: Engage with the community through open-source projects and reproducibility challenges.

Reproducibility Challenges in ML and Systems Research

Reproducibility Challenges in ML and Systems Research

- **Complexity of Experiments**
  - **Machine Learning**: Involves large datasets, complex models, and extensive hyperparameter tuning.
  - **Systems Research**: Requires intricate setups, hardware configurations, and environmental dependencies.
- **Barriers to Reproducibility**
  - **Insufficient Documentation**: Lack of detailed methodological descriptions.
  - **Proprietary Code and Data**: Closed-source code and inaccessible datasets hinder replication.
  - **Resource Constraints**: High computational costs make replication difficult for those without access to significant resources.

Reproducibility Initiatives in Machine Learning

NeurIPS Reproducibility Program

- **Background**
  - **Conference**: Neural Information Processing Systems (NeurIPS), one of the premier conferences in ML.
  - **Initiative Launch**: Recognizing reproducibility issues, NeurIPS introduced measures to encourage reproducibility.
- **Key Components**
  - **Reproducibility Checklist**
    - Authors are required to complete a checklist addressing:
      - Code availability.
      - Dataset details.
      - Experimental setup and hyperparameters.
  - **Code Submission Policy**
    - Strongly encourages authors to submit code alongside papers.
    - Code is reviewed to ensure it supports the claims made in the paper.
- **Impact**
  - **Increased Transparency**: More papers include code and detailed experimental setups.
  - **Community Engagement**: Encourages discussions around reproducibility and best practices.

Reproducibility Initiatives in Machine Learning

ICLR Reproducibility Challenge

- **Background**
  - **Conference**: International Conference on Learning Representations (ICLR).
  - **Purpose**: To engage the community in reproducing results from papers accepted at ICLR.
- **Process**
  - **Participation**
    - Open to students, researchers, and practitioners.
    - Participants select a paper from ICLR to replicate.
  - **Deliverables**
    - Replication report detailing the replication process, results, and any deviations from the original work.
  - **Publication**
    - Reports are published on platforms like OpenReview, promoting open access and discussion.
- **Benefits**
  - **Educational Value**: Provides hands-on experience in ML research.
  - **Improved Practices**: Feedback leads authors to enhance their documentation and code sharing.

Reproducibility Initiatives in Systems Research

ACM SIGPLAN Artifact Evaluation Process

- **Background**
  - **SIGPLAN**: Special Interest Group on Programming Languages.
  - **Artifact Evaluation**: Introduced to assess the availability, functionality, and reproducibility of research artifacts.
- **Process**
  - **Submission of Artifacts**
    - Authors submit code, data, and other artifacts alongside the paper.
  - **Evaluation Criteria**
    - **Functional**: Does the artifact work as described?
    - **Reusable**: Can others build upon it?
    - **Reproducible**: Can results be replicated using the artifact?
- **Recognition**
  - **Badges Awarded**
    - Papers receive badges indicating the level of artifact evaluation (e.g., "Artifact Available," "Results Reproduced").
  - **Incentive**
    - Recognizes and rewards authors who invest in making their work reproducible.

Reproducibility Initiatives in Systems Research

USENIX Reproducibility Initiative

- **Background**
  - **USENIX Association**: Hosts several top systems conferences like OSDI and ATC.
  - **Goal**: To promote reproducibility in experimental computer science.
- **Key Actions**
  - **Artifact Track**
    - Separate track for submitting artifacts associated with accepted papers.
  - **Best Reproducibility Award**
    - Acknowledges outstanding efforts in providing reproducible research.
- **Impact**
  - **Cultural Shift**: Encourages authors to prioritize reproducibility.
  - **Community Standards**: Establishes expectations for artifact sharing.

Best Practices for Reproducible Research

1. **Provide Detailed Methodologies**
   - **Experimental Setup**: Describe hardware, software versions, and configurations.
   - **Hyperparameters**: List all parameters used in training and evaluation.
2. **Share Code and Data**
   - **Open-Source Repositories**: Use platforms like GitHub or GitLab.
   - **Data Accessibility**: Ensure datasets are available or provide alternatives.
3. **Use Standardized Formats**
   - **Notebooks**: Share code using Jupyter Notebooks for transparency.
   - **Documentation**: Include README files with setup instructions.
4. **Automate Experimentation**
   - **Scripts for Reproduction**: Provide scripts to run experiments end-to-end.
   - **Environment Management**: Use tools like Docker or Conda for environment replication.
5. **Engage with the Community**
   - **Collaborate**: Encourage others to replicate and build upon your work.
   - **Respond to Feedback**: Be open to questions and ready to provide clarifications.

Tools and Resources Supporting Reproducibility

- **Version Control Systems**
  - **Git**: Track changes and collaborate on code.
  - **GitHub/GitLab**: Host repositories and manage projects.
- **Environment Management**
  - **Docker**: Containerization for consistent environments.
  - **Conda/Virtualenv**: Manage Python environments and dependencies.
- **Experiment Management Platforms**
  - **MLflow**: Track experiments, parameters, and results.
  - **Weights & Biases**: Monitor training runs and collaborate.
- **Data Sharing Platforms**
  - **Kaggle Datasets**: Find and share datasets.
  - **Zenodo**: Archive and share research outputs.

Impact of Reproducibility Initiatives

- **Improved Research Quality**
  - Encourages meticulous documentation and validation.
  - Leads to more robust and reliable findings.
- **Enhanced Collaboration**
  - Facilitates building upon others' work.
  - Accelerates innovation through shared knowledge.
- **Cultural Shift in the Community**
  - Establishes reproducibility as a standard expectation.
  - Promotes openness and transparency in research practices.

Conclusion

- **Reproducibility is Essential**
  - It is a cornerstone of scientific progress and integrity.
- **Active Participation Matters**
  - By engaging in reproducibility efforts, you contribute to the advancement of the field.
- **Ongoing Efforts**
  - The community continues to develop tools, policies, and cultures that support reproducibility.
- **Your Role**
  - As emerging researchers and practitioners, you have the power to shape the future of reproducible science.