Lecture 04 - Contents

An overview of the parts in the medical fine-tuning lecture.

This outline is for guidance. Navigate the slides with the left/right arrow keys.

Lecture 4:

Fine-tuning Strategies for Medical LLMs

Domain Adaptation

Ho-min Park

homin.park@ghent.ac.kr

powersimmani@gmail.com

Fine-Tuning Overview

Full Fine-Tuning vs Parameter-Efficient Methods
Adapting pre-trained models to medical domain

Full Fine-Tuning

- Updates all model parameters
- Requires large GPU memory
- Best performance on large datasets
- High computational cost

Parameter-Efficient (PEFT)

- Updates only small subset
- 10-100x less memory
- Faster training
- Good for limited resources

Medical Data Characteristics

- Limited labeled data
- High annotation cost
- Domain-specific terminology
- Privacy constraints

Resource Requirements

- GPU memory: 24-80GB
- Training time: hours to days
- Dataset size: 1K-1M samples
- Storage: 10-500GB

⚡ Key Considerations

- **Trade-off:** Performance vs. Efficiency vs. Resource availability
- **Strategy:** Start with PEFT methods, scale to full fine-tuning if needed
- **Safety:** Medical domain requires careful validation and testing

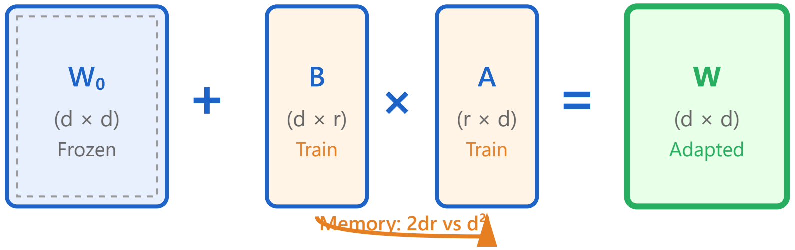
Part 1/3:

Parameter-Efficient Fine-Tuning (PEFT)

1. LoRA: Low-Rank Adaptation for Medical Applications
2. QLoRA: Memory Optimization with Quantization
3. Prefix Tuning Techniques
4. Adapter Modules
5. Parameter Selection Strategies
6. Rank Selection Guidelines
7. Memory-Compute Trade-offs

LoRA: Low-Rank Adaptation

Matrix Decomposition: $W = W_0 + BA$ where $\text{rank}(B) = \text{rank}(A) = r$



0.1% Trainable Params	**r=8-16** Typical Rank	**100x** Memory Reduction

Medical NER Application

- Disease entity recognition
- Symptom identification
- Medication extraction
- Procedure coding

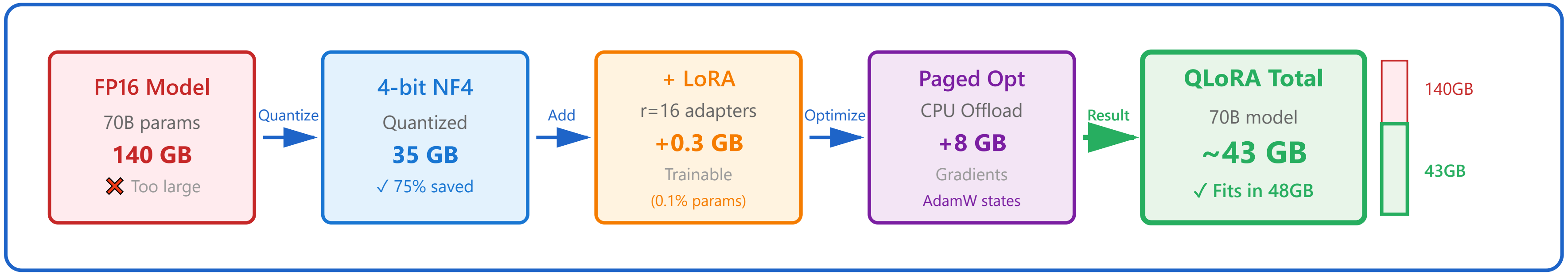
Configuration Guidelines

- **r=4:** Simple tasks, limited data
- **r=16:** Complex reasoning
- **r=8:** Most medical NLP tasks
- **r=32-64:** Multi-task learning

$$\text{Memory Saving} = 1 - (2 \times r \times d) / (d \times d) \approx 99.9\%$$

QLoRA: Quantized Low-Rank Adaptation

4-bit Quantization + LoRA
Train 70B models on single 48GB GPU



75%	<1%	24GB
Memory Reduction	Performance Loss	Min GPU Memory

✨ Key Benefits

- Enables large model fine-tuning
- Reduces training costs
- Maintains model accuracy
- Faster convergence

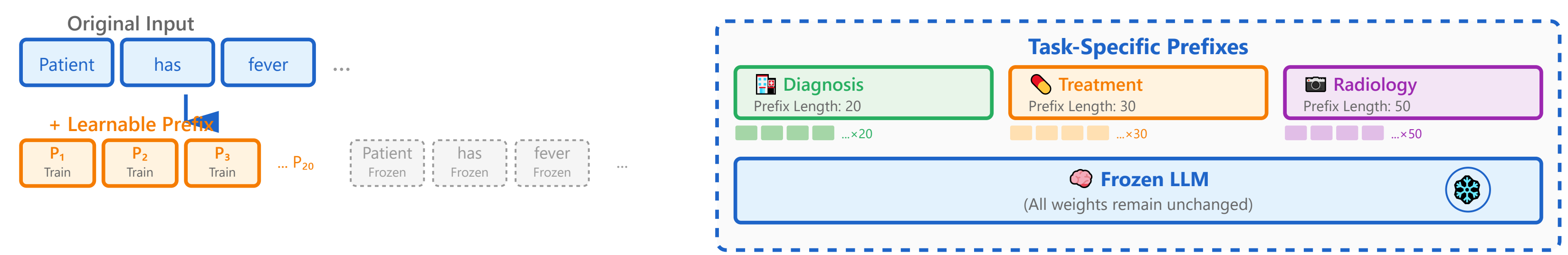
Implementation Details

- NF4 (Normal Float 4-bit)
- Paged optimizers
- Double quantization
- BitsAndBytes library

Prefix Tuning Techniques

Learnable Prefix Vectors

Prepend trainable tokens to input without modifying model weights



0.01% Trainable Params	**10-100** Prefix Length	**Fast** Training Speed

🎯 Medical Prompt Design

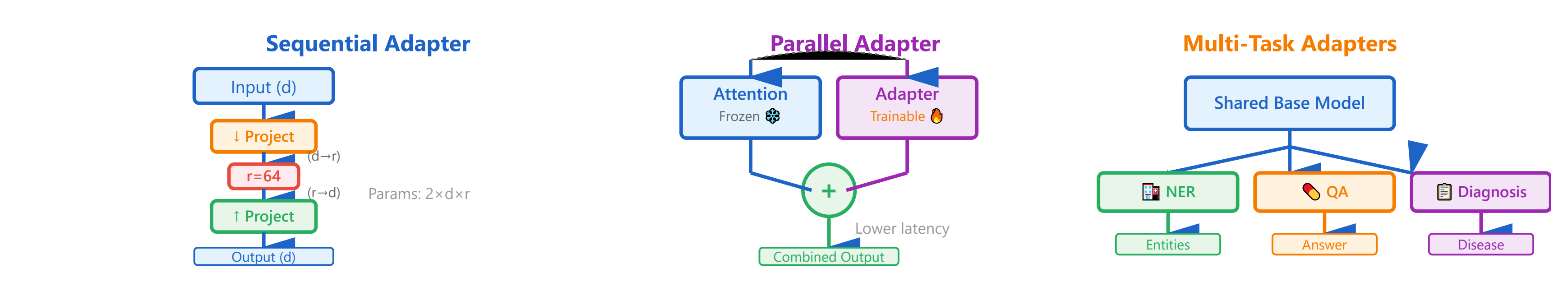
- Task-specific prefixes (diagnosis, treatment, education)
- Specialty-specific prefixes (radiology, pathology)
- Multi-task prefix sharing
- Prefix length optimization (20-50 typical)

⚡ Advantages

- Minimal parameter updates
- Easy to switch between tasks
- No architectural changes
- Composable prefixes

Adapter Modules

Insert Small Layers between frozen model layers
Parallel vs Sequential adapter architectures



1-5%	64-512	Modular
Added Params	Bottleneck Size	Architecture

🔧 Adapter Types

- **Sequential:** Down-project → ReLU → Up-project
- **Parallel:** Run alongside attention layers
- **Multi-task:** Separate adapters per task
- **Stacked:** Layer-wise adapter composition

🥼 Medical Task Adapters

- Clinical NER adapter
- Medical QA adapter
- Diagnosis prediction adapter
- Report generation adapter

Parameter Selection Strategies

Learning Rate Scheduling & Layer-wise Optimization
Critical hyperparameters for successful fine-tuning

📊 Learning Rate Guidelines

- **PEFT:** 1e-3 to 5e-3 (higher than full fine-tuning)
- **Full fine-tuning:** 1e-5 to 5e-5
- **Warmup:** 5-10% of total steps
- **Decay:** Linear or cosine to 10% of peak

⚙️ Layer-wise Learning Rates

- Lower layers: Smaller LR (preserve general features)
- Upper layers: Higher LR (adapt to domain)
- Ratio: 1:10 (bottom:top layers)
- Gradual unfreezing strategy

🛡️ Regularization Techniques

- Weight decay: 0.01-0.1
- Dropout: 0.1-0.3 in adapters
- Early stopping: patience 3-5 epochs

- Gradient clipping: max norm 1.0

Rank Selection Guidelines

LoRA Rank Selection: r=1~64
Balance between performance and efficiency

r=4-8	r=16-32	r=64+
Simple Tasks	Most Tasks	Complex Tasks

📊 Task Complexity Recommendations

- **r=4:** Binary classification, sentiment analysis
- **r=8:** Named entity recognition, simple QA
- **r=16:** Medical report generation, diagnosis
- **r=32:** Clinical reasoning, multi-task learning
- **r=64:** Large-scale medical knowledge integration

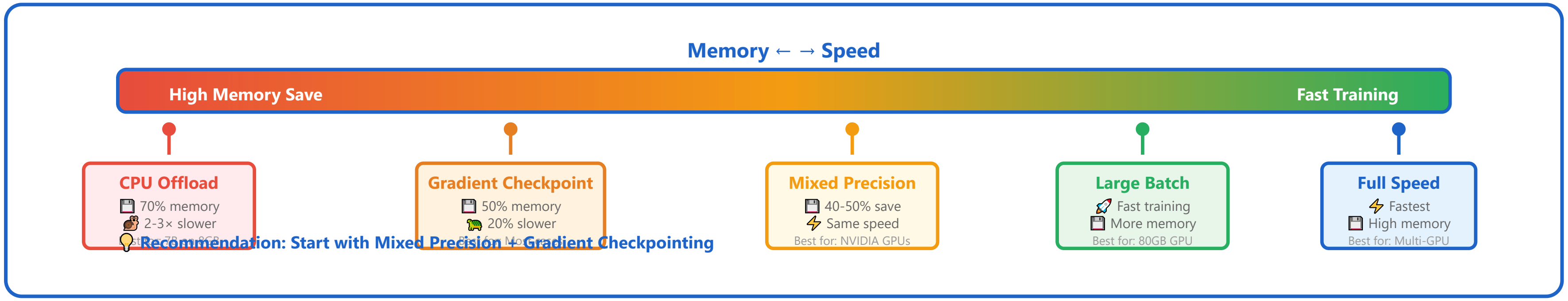
🔬 Experimental Approach

- Start with r=8, measure validation performance
- Double rank if performance plateaus early
- Halve rank if overfitting occurs
- Monitor training loss vs validation gap

Memory-Compute Trade-offs

GPU Memory vs Training Time

Optimization strategies for limited resources



💾 Memory Optimization Techniques

- **Gradient Checkpointing:** 50% memory, 20% slower
- **Batch Size Reduction:** Linear memory scaling
- **Gradient Accumulation:** Effective batch size increase
- **Mixed Precision:** 40-50% memory reduction
- **CPU Offloading:** 70% memory, 2-3x slower

⚡ Recommended Configurations

- **24GB GPU:** 7B models with QLoRA, batch=4
- **40GB GPU:** 13B models with QLoRA, batch=8
- **80GB GPU:** 30B models with LoRA, batch=16

- **Multi-GPU:** 70B models with DeepSpeed ZeRO-3

📈 Resource Planning

- Profile memory usage before full training run
- Leave 20% GPU memory as buffer
- Monitor CUDA OOM errors and adjust

Part 2/3:

Medical Instruction Tuning

1. Medical Instruction Datasets
2. Clinical Task Formulation
3. Multi-Task Learning
4. Curriculum Learning for Medical Domain
5. Catastrophic Forgetting Prevention
6. Domain Mixture Strategies

Medical Instruction Datasets

Public Medical Instruction Datasets
High-quality data for medical LLM training

📚 Major Datasets

- **MedInstruct:** 100K clinical instructions (diagnosis, treatment)
- **HealthCareMagic:** 200K patient-doctor conversations
- **MedQA:** 60K medical exam questions with explanations
- **PubMedQA:** 1K expert-annotated biomedical QA pairs
- **ChatDoctor:** 100K patient-physician dialogues

40%	30%	30%
Diagnosis	Treatment	Education

✅ Quality Metrics

- Expert validation rate: 85-95%
- Average response length: 150-300 tokens
- Domain coverage: 50+ medical specialties
- Language diversity: English, Chinese, Spanish

Clinical Task Formulation

Converting Clinical Tasks to Instructions
Structured templates for medical workflows

🏥 Task Templates

- **Diagnosis:** "Given symptoms: [X], provide differential diagnosis"
- **Treatment:** "For patient with [condition], recommend treatment plan"
- **Test Interpretation:** "Analyze lab results: [values], clinical significance"
- **Patient Education:** "Explain [condition] in patient-friendly language"

📋 Input-Output Format

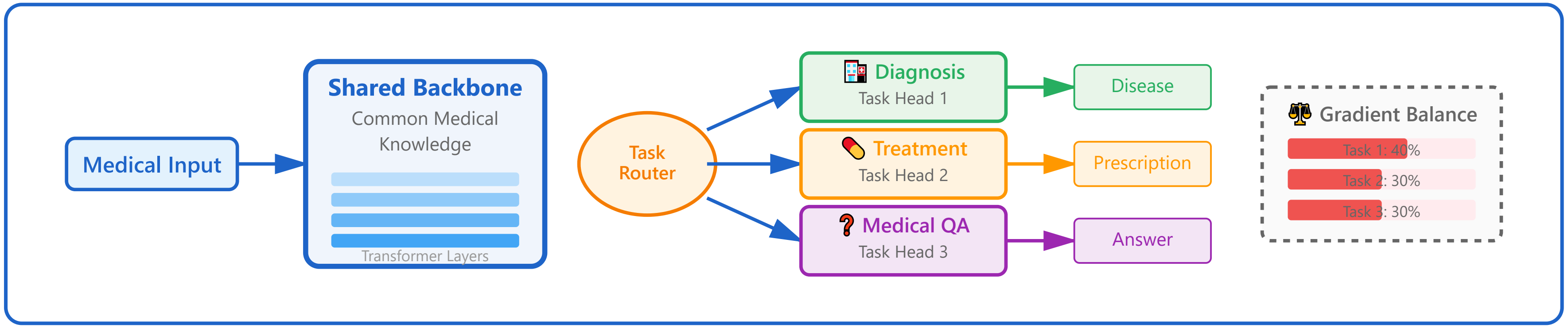
- **Input:** Patient context + Clinical question
- **Output:** Structured medical response + Reasoning
- Include confidence scores where appropriate
- Add safety disclaimers for critical decisions

🔄 Conversion Examples

- EHR → "Summarize patient history for consultation"
- Radiology → "Describe findings in chest X-ray"
- Drug info → "List contraindications for [medication]"

Multi-Task Learning

Simultaneous Training on Multiple Medical Tasks
Share representations while preventing negative transfer



⚖️ Task Weighting Strategies

- **Uniform:** Equal weight for all tasks
- **Loss-based:** Weight by inverse of loss magnitude
- **Uncertainty:** Weight by task uncertainty/difficulty
- **Dynamic:** Adjust weights during training

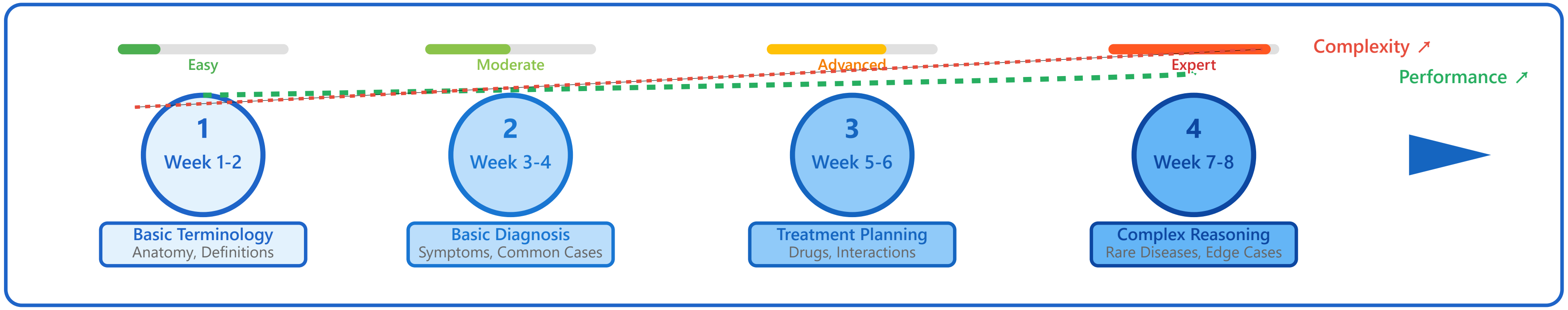
🚫 Preventing Negative Transfer

- Monitor per-task validation performance
- Use task-specific batch normalization
- Implement task clustering (group similar tasks)

- Apply gradient surgery techniques

Curriculum Learning for Medical Domain

Easy → Difficult Task Progression
Hierarchical medical knowledge acquisition



🎓 Knowledge Hierarchy

- **Level 1:** Facts (drug names, normal lab values)
- **Level 2:** Concepts (disease mechanisms, physiology)
- **Level 3:** Reasoning (differential diagnosis)
- **Level 4:** Clinical judgment (treatment selection)

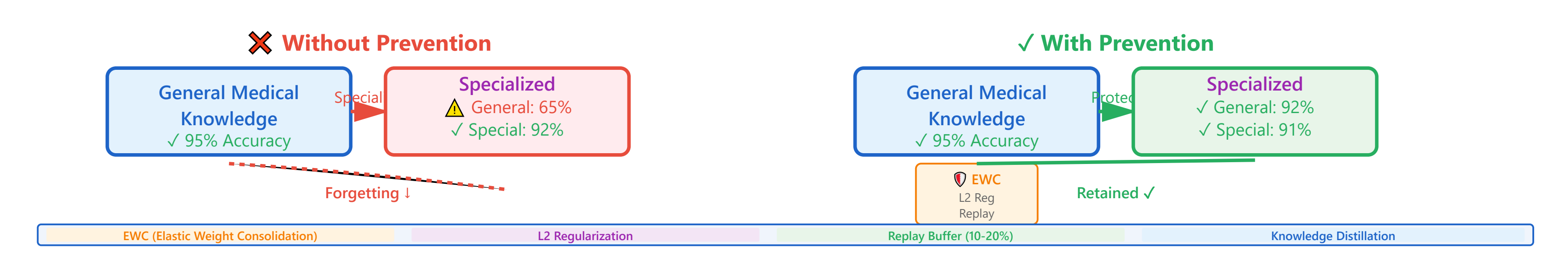
📈 Performance Benefits

- 10-15% accuracy improvement on complex tasks
- Faster convergence (30% fewer epochs)
- Better knowledge retention

- More stable training dynamics

Catastrophic Forgetting Prevention

Preserve General Medical Knowledge
While adapting to specialized domains



90%+ Knowledge Retention	**10-20%** Replay Data	**λ=0.1-1.0** EWC Penalty

🛡 Prevention Techniques

- **Elastic Weight Consolidation (EWC):** Protect important parameters
- **L2 Regularization:** Stay close to pre-trained weights
- **Replay Buffer:** Mix old and new data (10-20% ratio)
- **Knowledge Distillation:** Maintain original model behavior

📊 Monitoring Strategy

- Track performance on general medical benchmarks
- Measure forgetting rate: Δ = Acc_initial - Acc_final
- Set threshold: Stop if forgetting > 5%
- Periodic evaluation on held-out general tasks

Domain Mixture Strategies

General + Medical Data Mixing Ratio
Optimal balance for domain adaptation

⚖️ Mixing Ratios

- **90% Medical / 10% General:** Strong specialization
- **80% Medical / 20% General:** Recommended baseline
- **70% Medical / 30% General:** Preserve general capabilities
- **50/50:** Multi-domain applications

80/20	+12%	-2%
Optimal Ratio	Medical Accuracy	General Accuracy

🎯 Domain Adaptation Levels

- **High Adaptation:** Clinical specialists (95% medical)
- **Moderate:** Medical assistants (80% medical)
- **Low:** General health chatbots (60% medical)

🔄 Dynamic Mixing

- Start with 50/50, gradually increase medical data

- Monitor both domain performances
- Adjust based on validation metrics

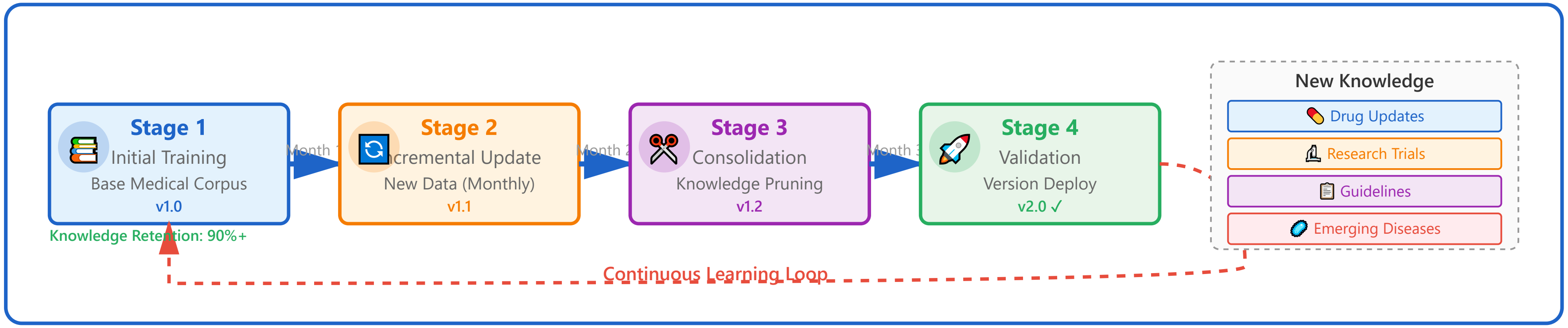
Part 3/3:

Advanced
Fine-Tuning Methods

1. Continual Learning Frameworks
2. Few-Shot Fine-Tuning
3. Reinforcement Learning from Human Feedback
4. Adversarial Training
5. Data Efficiency Techniques
6. Hyperparameter Optimization

Continual Learning Frameworks

Continuous Integration of New Medical Knowledge
Lifelong learning without retraining from scratch



Version Management

- Model versioning: v1.0, v1.1, v2.0 (semantic)
- Track knowledge cutoff dates
- Maintain backward compatibility
- A/B testing for new versions

New Knowledge Integration

- **Drug updates:** New medications and guidelines
- **Research:** Latest clinical trial results
- **Guidelines:** Updated treatment protocols

- **Epidemiology:** Emerging diseases and trends

Few-Shot Fine-Tuning

Learn from 5-100 Examples
Critical for rare diseases and limited data scenarios

5-10	10-100	70-85%
Few-Shot	Low-Resource	Accuracy

🏥 Rare Disease Applications

- Orphan diseases with < 100 documented cases
- Novel disease presentations (e.g., COVID-19 variants)
- Specialized diagnostic criteria
- Unique patient populations

🎯 Few-Shot Strategies

- **Meta-Learning:** Learn to learn from few examples
- **Prompt-Based:** Design task-specific prompts
- **Data Augmentation:** Synthetic example generation
- **Transfer Learning:** From similar rare diseases

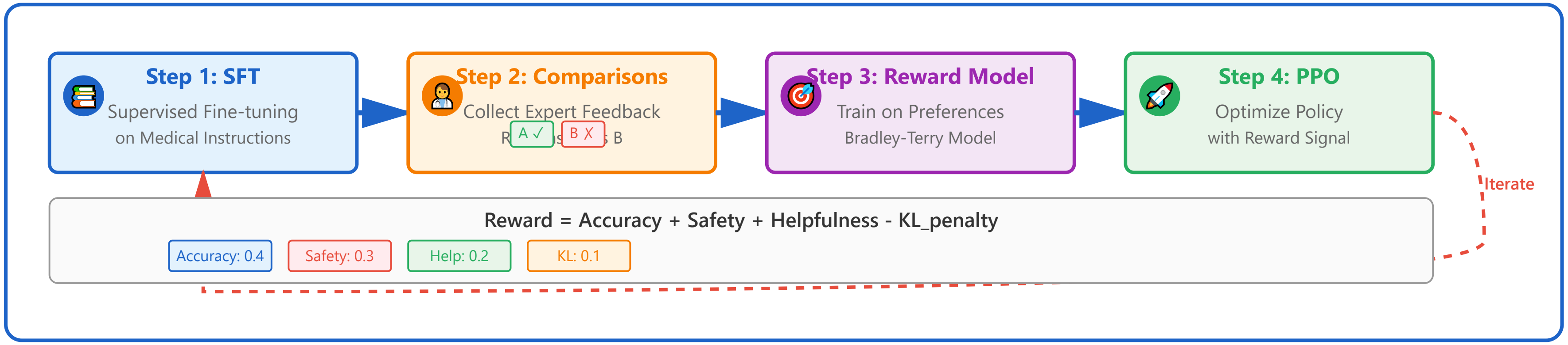
⚙️ Implementation Tips

- Use higher learning rates (5e-3 to 1e-2)
- Train for more epochs (20-50 vs 3-5)
- Apply strong regularization
- Validate with leave-one-out cross-validation

Reinforcement Learning from Human Feedback

RLHF for Medical Domain

Align model outputs with medical expert preferences



10K+ Comparisons	**PPO** Algorithm	**+15%** Expert Preference

Medical Expert Feedback

- **Accuracy:** Factual correctness of medical information
- **Safety:** Avoidance of harmful recommendations
- **Clarity:** Patient-friendly explanations
- **Completeness:** Comprehensive coverage of topic

Adversarial Training for Robustness

Generate Adversarial Examples
Improve model robustness and safety

🎯 Adversarial Strategies

- **Perturbation:** Add noise to medical terms
- **Synonym Replacement:** Use alternative medical terminology
- **Paraphrasing:** Rephrase clinical questions
- **Context Variation:** Change patient demographics

🛡️ Safety Enhancement

- Detect and prevent harmful outputs
- Improve consistency across input variations
- Reduce sensitivity to typos and misspellings
- Handle ambiguous medical queries

20-30%	95%+	FGSM
Robustness Gain	Safety Score	Attack Method

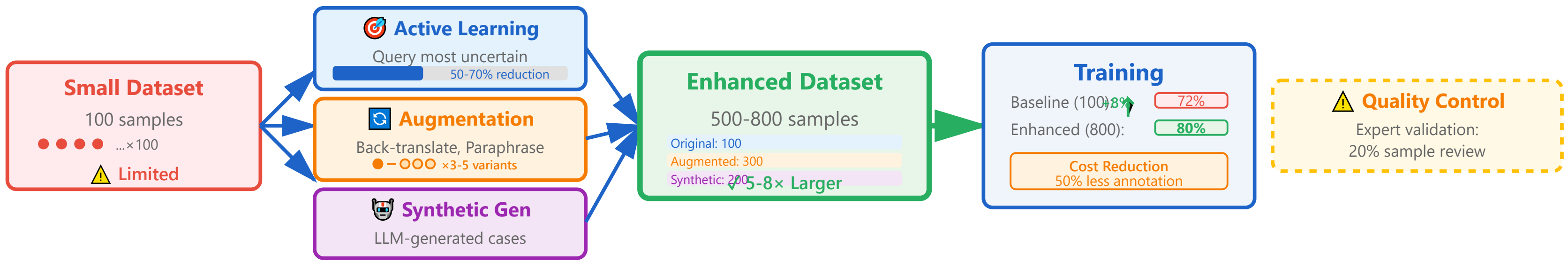
🔬 Testing Protocol

- Generate 1000+ adversarial test cases
- Measure accuracy degradation under attack
- Ensure consistent medical reasoning

Data Efficiency Techniques

Maximize Learning from Limited Medical Data

Active learning, augmentation, and synthesis



3-5x	50%	+8%
Data Efficiency	Cost Reduction	Accuracy Gain

🎯 Active Learning

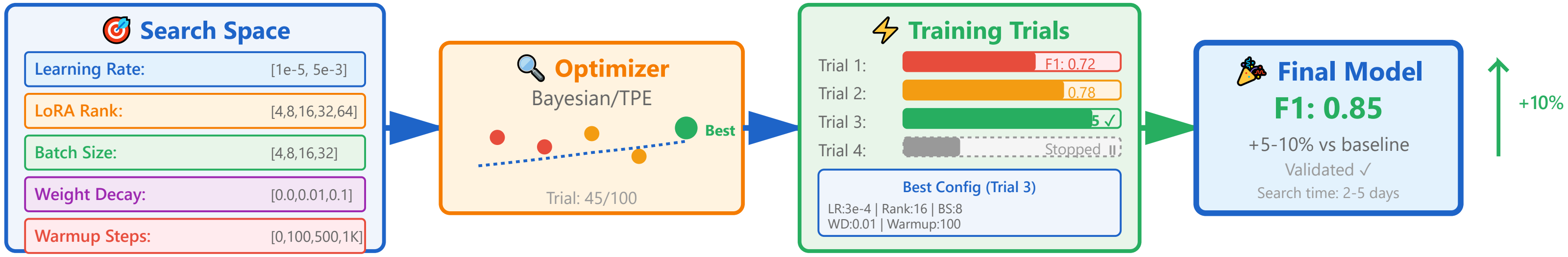
- **Uncertainty Sampling:** Select most uncertain predictions
- **Diversity Sampling:** Cover different case types
- **Query-by-Committee:** Multiple model disagreement
- Reduce annotation needs by 50-70%

Data Augmentation

- **Back-translation:** Translate to another language and back
- **Synonym replacement:** Medical term substitution
- **Paraphrasing:** GPT-4 based rephrasing
- **Mixup:** Interpolate between examples

Hyperparameter Optimization

Automated Hyperparameter Search
Find optimal configurations for medical tasks



50-100	5-10%	2-5 days
Trials	Performance Gain	Search Time

Optimization Tools

- **Optuna:** Tree-structured Parzen estimator (TPE)
- **Ray Tune:** Distributed hyperparameter tuning
- **Weights & Biases Sweeps:** Bayesian optimization
- **Hyperopt:** Random search and TPE

📊 Search Strategies

- **Random Search:** 20-50 trials, good baseline
- **Bayesian Optimization:** 50-100 trials, efficient
- **Population-Based:** Evolve configurations
- **Early Stopping:** Terminate poor trials

Evaluation During Training

Continuous Monitoring & Validation
Early detection of overfitting and convergence issues

📊 Validation Strategy

- **Frequency:** Every 100-500 steps or 1 epoch
- **Validation Set:** 10-20% of training data
- **Stratification:** Balanced across medical specialties
- **Time-based Split:** Avoid data leakage

🎯 Medical Metrics

- **Accuracy:** Overall correctness
- **F1 Score:** Balance precision and recall
- **Medical Entity F1:** NER performance
- **Clinical Relevance:** Expert judgment scores
- **Safety Metrics:** Harmful output rate

🛑 Early Stopping Criteria

- **Patience:** 3-5 epochs without improvement
- **Delta:** Min improvement threshold (0.001-0.01)
- **Monitor:** Validation loss or task-specific metric

- **Restore:** Best checkpoint on early stop

Every 500	Patience=3	5-10
Steps	Epochs	Metrics

Model Selection Criteria

Choosing the Right Model
Performance, efficiency, safety, and regulatory compliance

⚖️ Selection Dimensions

- **Performance:** Accuracy on medical benchmarks
- **Efficiency:** Inference latency and throughput
- **Safety:** Error rate on critical tasks
- **Interpretability:** Explanation capabilities
- **Cost:** Training and deployment expenses

🏥 Medical Safety Requirements

- Sensitivity to harmful outputs: < 0.1%
- Critical error rate: < 1%
- Consistency across similar inputs: > 95%
- Appropriate uncertainty quantification

📋 Regulatory Considerations

- **FDA:** Software as Medical Device (SaMD)
- **HIPAA:** Patient data privacy compliance
- **EU MDR:** Medical device regulation

- **Documentation:** Model cards, risk analysis

🚀 Deployment Constraints

- **Latency:** < 2s for interactive applications
- **Hardware:** Available GPU/CPU resources
- **Scalability:** Handle concurrent users

Case Study: Radiology Report Generation

Chest X-ray Report Generation
Fine-tuning LLaMA-7B with LoRA on MIMIC-CXR

📊 Dataset: MIMIC-CXR

- **Size:** 377,110 chest X-ray images + reports
- **Split:** 80% train, 10% val, 10% test
- **Input:** Image features + patient demographics
- **Output:** Findings + Impressions sections

🔧 LoRA Configuration

- **Base Model:** LLaMA-7B
- **Rank:** r=16
- **Alpha:** α=32
- **Target Modules:** q_proj, v_proj
- **Learning Rate:** 3e-4
- **Batch Size:** 16 (gradient accumulation=4)

BLEU-4	ROUGE-L	Training
0.176 → 0.213	0.348 → 0.392	8 hours (A100)

✅ Results & Insights

- 22% improvement in BLEU-4 score
- Better clinical terminology usage
- Reduced hallucinations (18% → 7%)
- Radiologist feedback: 85% acceptable quality

Hands-On: Fine-Tuning Implementation

Practical Implementation with Hugging Face PEFT
Step-by-step guide to medical LLM fine-tuning

🔧 Setup & Installation

- **Libraries:** transformers, peft, datasets, bitsandbytes
- **Hardware:** Single GPU with 24GB+ memory
- **Environment:** Python 3.9+, PyTorch 2.0+, CUDA 11.8+

💻 Code Implementation

- **Step 1:** Load pre-trained model (LLaMA-7B, Mistral-7B)
- **Step 2:** Configure LoRA (rank=8, alpha=16)
- **Step 3:** Prepare medical dataset (tokenization)
- **Step 4:** Set training arguments (epochs=3, lr=3e-4)
- **Step 5:** Train with Trainer API
- **Step 6:** Merge LoRA weights and save

📈 Monitoring & Logging

- **TensorBoard:** Track loss curves and metrics
- **W&B:** Experiment tracking and comparison

- **GPU Monitoring:** nvidia-smi, watch -n1

30 min	2-8 hrs	100 lines
Setup Time	Training Time	Code

Best Practices Summary

10 Key Best Practices
Lessons learned from medical LLM fine-tuning

✅ Do's

- **Start Small:** Begin with PEFT methods (LoRA r=8)
- **Validate Extensively:** Use multiple medical benchmarks
- **Monitor Safety:** Track harmful output rates continuously
- **Use Mixed Precision:** FP16 for memory efficiency
- **Document Everything:** Model cards, training logs

❌ Don'ts

- **Don't Skip Validation:** Always use held-out test sets
- **Don't Ignore Forgetting:** Monitor general capabilities
- **Don't Over-tune:** Stop at validation peak
- **Don't Deploy Untested:** Require expert review
- **Don't Forget Privacy:** De-identify training data

🔍 Common Mistakes

- Using too high learning rates (causes instability)
- Insufficient data preprocessing (noise in medical texts)

- Ignoring class imbalance (rare diseases underrepresented)
- Not using gradient accumulation (memory constraints)

📋 Pre-Deployment Checklist

☑ Performance validated on multiple test sets

☑ Safety metrics within acceptable thresholds

☑ Expert review completed (3+ medical professionals)

☑ Regulatory compliance documented

Thank You!
Master PEFT techniques for medical LLM adaptation
Apply instruction tuning and advanced methods
Deploy safe and effective medical AI systems
📚 Hugging Face PEFT Documentation
🔬 MedInstruct Dataset on GitHub
💬 Join Medical AI Community
Questions?
homin.park@ghent.ac.kr | powersimmani@gmail.com