Machine Learning Optimization Methods and Their Application to Predicting T-Cell Differentiation Using GNNs and Reinforcement Learning

Francis Boabang

Link to source codes

https://github.com/boabangf/GNN_RL_gene_trajectory_perturbation/tree/main

Content of Seminar

- Presentation of 2024 Ph.D. Thesis
- Application to Stem Cell Differentiation

Refining Optimization Methods for Training Machine Learning Models: A Case Study in Robotic Surgical Procedures

2024 Ph.D. Thesis

Francis Boabang

Link to thesis:

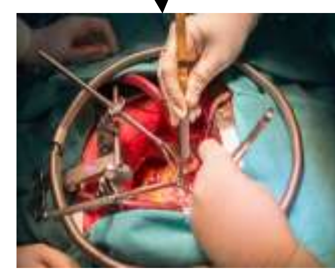
https://spectrum.library.concordia.ca/id/eprint/994877/

Content

- Introduction
- Challenges
- Motivation
- Thesis Statement
- Contribution to the Thesis
- First Contribution
- Second Contribution
- Conclusion of the Thesis
- Limitations
- Future Work

Introduction

A surgeon in a particular geographical location manipulate the surgeon console to operate on a remote patient



Open heart surgery

- Benefits of Remote Robotic Surgery

✓ Improves surgeons' precision
✓ Reduce patient blood loss
✓ Ensures shorter hospital stay

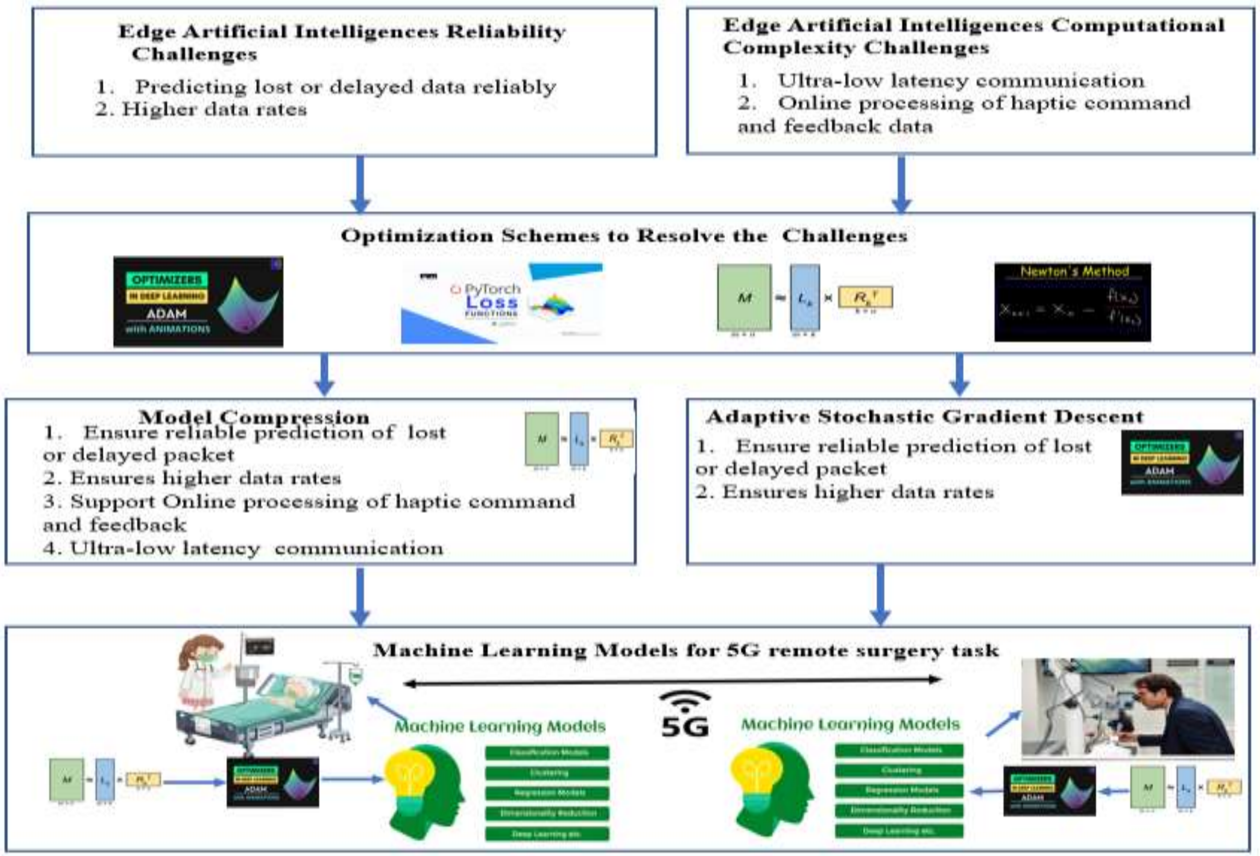
- Examples of Remote Robotic Surgery

✓ Facial reconstruction surgery
✓ Knot-tying
✓ Suturing
✓ Needle Passing
✓ Trocar Insertion

Chowriappa, A., Wirz, R., Ashammagari, A.R. *et al.* **Prediction from expert demonstrations for safe tele-surgery**. *Int. J. Autom. Comput.* 10, 487–497 (2013)

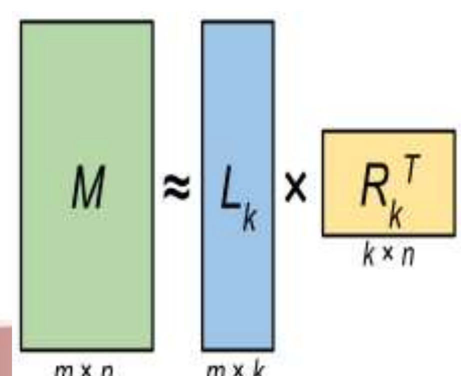
Challenges

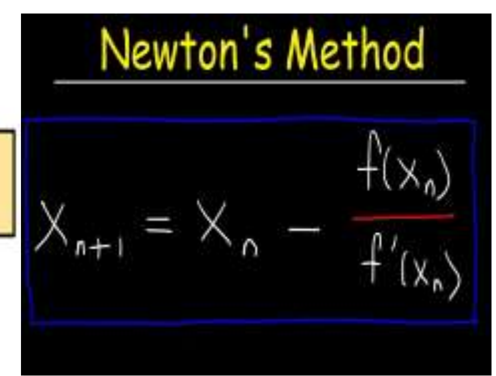
- Accurate Prediction
- Ultra-low latency communication
- Online processing of haptic command and feedback



Thesis Statement

- We have proposed enhanced stochastic gradient descent and low-rank matrix factorization for training machine learning models
- The improvements focus on ensuring the convergence of these optimization schemes during machine learning training
- These two optimization schemes collaborate, rather than compete, to enhance machine learning reliability for remote robotic surgery procedures



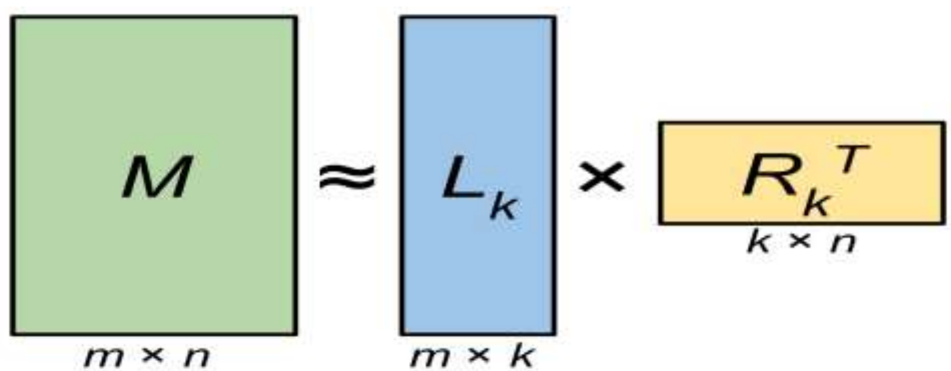


Contribution of the Thesis

- An Improved Low Rank Matrix Factorization to Scale Machine Learning and Its Application to Remote Robotic Surgery [Published in IEEE TNSM]

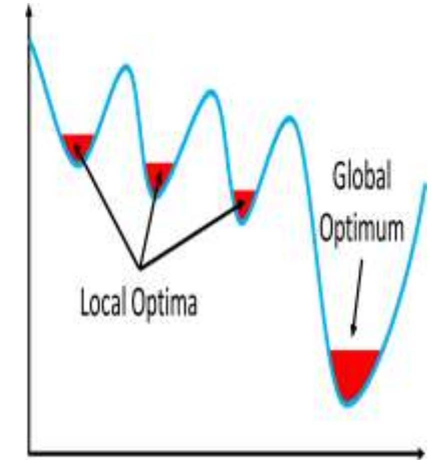
- Enhanced Stochastic Gradient Descent Algorithm for Machine Learning Training and Its Application in Remote Surgical Gesture Recognition [Revising, IEEE transactions on Neural Network and Learning Systems]

- An Improved Low Rank Matrix Factorization to Scale Machine Learning and Application to Remote Robotic Surgery [Published in IEEE TNSM]



Francis Boabang, A. Ebrahimzadeh, R. Glitho, H. Elbiaze, M. Maier and F. Belqami, "**A Machine Learning Framework for Handling Delayed/Lost Packets in Tactile Internet Remote Robotic Surgery**," in IEEE Transactions on Network and Service Management, doi: 10.1109/TNSM.2021.3106577

Review of Machine Learning Model Compression Techniques



- **Low rank and sparse matrix factorization**
- ✓ It can be used to scale ML methods and address delay and lost data online

- **Types of low rank and sparse matrix factorization**
- ✓ Nonconvex formulation of low rank and sparse matrix factorization
- ✓ Convex formulation of low rank and sparse matrix factorization

- **Challenges of nonconvex formulation of low rank and sparse matrix factorization**

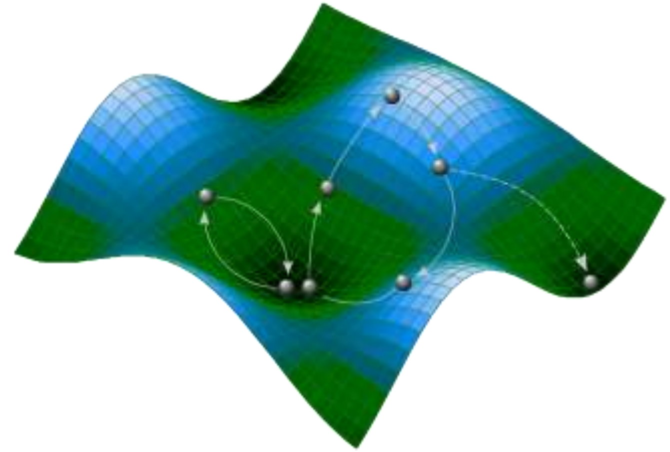
- ✓ Non-convex low-rank matrix factorization achieves higher accuracy with less computation compared to convex models
- ✓ Non-convex formulations struggle to find quality solutions without good initial points

- ✓ The hybrid model (convex and non-convex) can converge to quality solutions even with random initialization

- ✓ The hybrid model is computationally efficient

Xiong L, Chen X, Schneider J. **Direct robust matrix factorization for anomaly detection**. In2011 IEEE 11th International Conference on Data Mining 2011 Dec 11 (pp. 844-853). IEEE.

Overview of the Low-Rank Approximation



- Nonconvex low-rank approximation

$$\|G - U\Sigma U^T\|_F,$$
$$s.t$$
$$rank\ (L) \leq r$$

- In many applications involving nonconvex machine learning methods, the upper bound of the rank r is known, rather than the exact $rank\ (L)$

$$\|G - U\Sigma U^T\|_F,\quad s.t\quad rank\ (L) \leq r$$

  - Convex relaxed low rank approximation

$$\|G - U\Sigma U^T\|_1 + \|\Sigma\|_*,$$
$$s.t$$
$$UU^T = I$$

  - $rank\ (L) \leq r$ is avoided in the convex relaxed low rank approximation

Related Work

Method	Mode	Complexities	Low Rank Matrix Recovery Rate
GMM/GMR based	Batch (offline)	$O(mk)$	N/A
HMM/GMR based	Batch	$O(mk)$	N/A
LWPR	Incremental (online)	$O(m)$	N/A
ELM based	Batch	$O(mh)$	N/A
KNNR based	Instance Based	$O(m^k)$	N/A
RL based	Batch	$O(m^3)$	N/A
DeepRL based	Batch	$O(m^3h)$	N/A
LSTM based	Batch	$O(m^3h)$	N/A
GPR based	Batch	$O(m^3)$	N/A
PCA GPR	Incremental	$O(r^2m)$	$O\left(\frac{1}{\varrho^2 r\beth}\right)$
RPCA GPR	Incremental	$O(r^2m)$	$O\left(\frac{1}{r\beth^2}\right)$
Proposed	Incremental	$O(r^2m)$	$O\left(\frac{1}{r\beth}\right)$

Proposed Method for the Covariance Matrix Update in Gaussian Process Regression

Initialization

$$\|G - U\Sigma U^T\|_1 + \|\Sigma\|_*$$
$$s.t$$
$$UU^T = I$$

Convex Relaxed Low rank Approximation or Mean square error regularized

$$rank\ (L)$$

$$\|G - U\Sigma U^T\|_F$$
$$s.t$$
$$rank\ (L) \leq r$$

Nonconvex Low Rank Approximation

We first compute rank (L) using convex relaxed low-rank approximation and used to initialize nonconvex low-rank approximation

Proposed Solution (Bayesian View)

Convex Initialization of low rank matrix factorization

Algorithm 1 Sequential ℓ_1 Norm Regularized Randomized Low-Rank and Sparse Matrix Factorization

1: Input: G: the kernel matrix, rank r, $\alpha = 10^{-3}$, $\beta = 10^{-5}$, and $\rho = 2$
2: j: iterative step
3: $j = 1$, $U_1 = \Sigma_1 = \hat{\Sigma}_1 = P_1 = 0$, $\beta_{max} = 10^{10}$, eigendecomposition$(G_j) = U_1 \Sigma_1 U_1^T$
4: Output: U, Σ, $\hat{\Sigma}$, P
5: **WHILE** not converged
6: **WHILE** not converged

$$\left\{\begin{array}{l} \text{Update } \Sigma_{j+1} \text{ by (12)}, \\ \text{Update } U_{j+1} \text{ by (14)}, \\ \text{Update } \hat{\Sigma}_{j+1} \text{ by (15)}, \\ \text{Update } P_{j+1} \text{ by (16)}, \\ \text{Update eigen decomposition of} \\ G_{j+1} \text{ by applying Theorem 1.} \end{array}\right.$$

7: **ENDWHILE**

$$\left\{\begin{array}{l} \text{Update the Lagrange multiplier} \\ \quad \Lambda_1 \text{ and } \Lambda_2 \text{ by (17)}, \\ \text{Update } \beta = \min(\rho\beta, \beta_{max}) \\ j = j + 1; \end{array}\right.$$

8: **ENDWHILE**

Nonconvex low rank matrix factorization

Algorithm 2 Sequential Randomized Low-Rank and Sparse Matrix Factorization

1: Input: G: the kernel matrix
2: γ: the maximal acceptable delay/ packet loss
3: S: matrix of delay/packet loss
4: j: iterative step
5: $j = 1$, $S_j = S$, $C_j = G_j - S$, eigen decomposition $(C_j) = U_1 \Sigma_1 U_1^T$
6: Output: U, Σ, S
7: **WHILE** not converged

$$\left\{\begin{array}{l} \Sigma_{j+1} = \arg\min\limits_{\Sigma} ||C_j - U_j \Sigma_j U_j^T||_F, \\ U_{j+1} = \arg\min\limits_{U} ||C_j - U_j \Sigma_j U_j^T||_F, \\ E_j = G_j - U_{j+1}\Sigma_{j+1}U_{j+1}^T, \\ S_{j+1} = \arg\min\limits_{card(S)\leqslant\gamma} ||E_j - S||_F, \\ \text{Update eigen decomposition of} \\ C_{j+1} \text{ by applying Theorem 1.} \end{array}\right.$$

8: $j = j+1$;
9: **ENDWHILE**

Machine Learning model: Gaussian Process Regression

Algorithm 3 Sequential Robust Randomized Gaussian Process Regression

1: Input: For $j = 1$, compute the kernel matrix G_1 using a kernel function and the intial input $\phi_1^{(d)}$
2: Compute U_1, Σ_1 using Algorithm 1
3: **for** $j \in \{2, 3, \ldots\ldots J\}$ **do**
4: Compute cross covariances $G(\phi^d, \phi^{d*})$ and $G(\phi^{d*}, \phi^{d*}) + \sigma_n^2 I$
5: Form

$$\bar{G}_j := \begin{bmatrix} U_{j-1}\Sigma_{j-1}U_{j-1}^T & G\left(\phi_j^{(d)}, \phi_j^{(d*)}\right) \\ G\left(\phi_j^{(d*)}, \phi_j^{(d)}\right) & G\left(\phi_j^{(d*)}, \phi_j^{(d*)}\right) + \sigma_n^2 I \end{bmatrix}$$

6: Using Algorithm 1 or 2, compute U_j, Σ_j
7: Compute ϑ^{-1} by Eq. (7)
8: Predict outputs for the new input $\phi_j^{(d)}$ using Eqs. (5) and (6)
9: **end for**

Predicting packet loss and delay data

Gesture

g_1^d $\cancel{g_2^d}$ g_3^d

Concordia University

Kinematic Dataset of Surgical Task

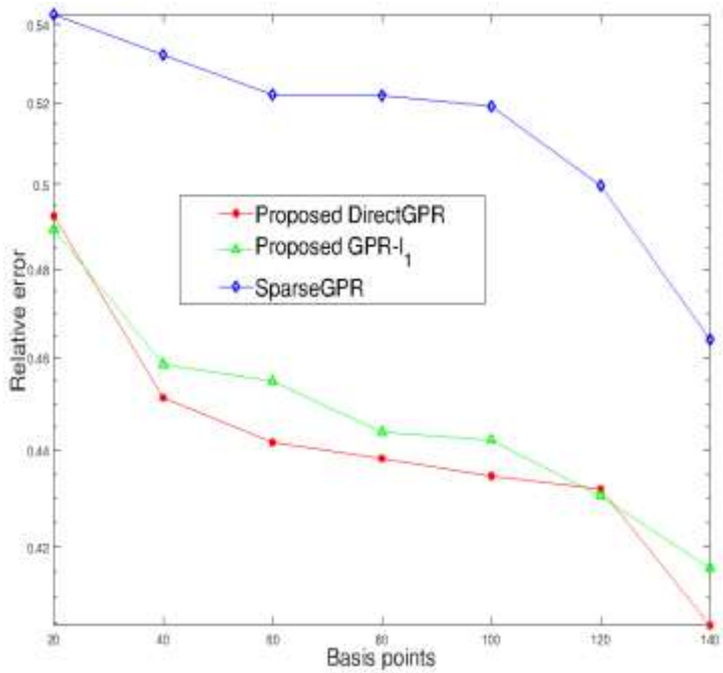
Suturing Knot-Tying Needle passing

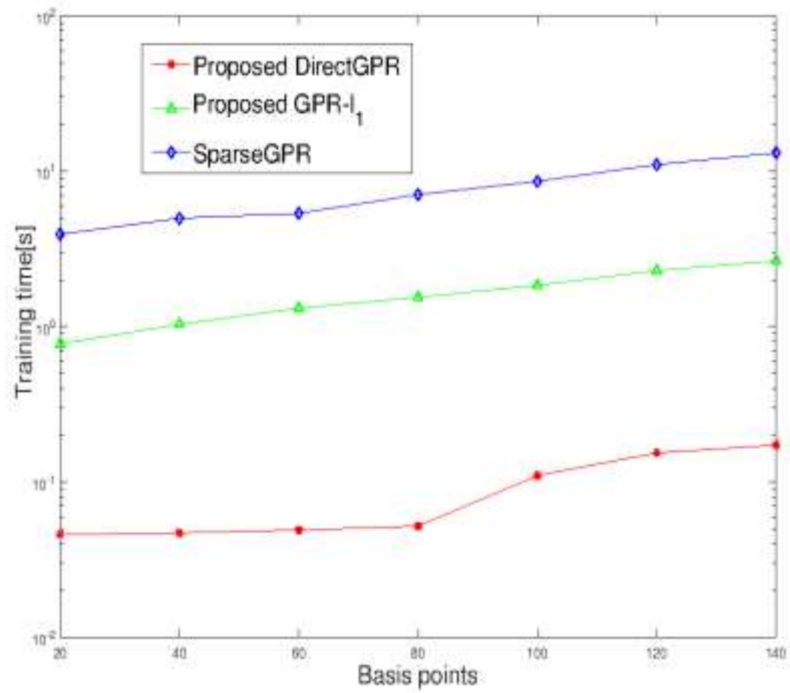
Column Indices	Number of Variables	Descriptions of Variables
1-3	3	Left MTM tool tip position (xyz)
4-12	9	Left MTM tool tip rotation matrix (R)
13-15	3	Left MTM tool tip linear velocity ($x'y'z'$)
16-18	3	Left MTM tool tip rotational velocity ($\alpha'\beta'\gamma'$)
19	1	Left MTM gripper angle velocity (θ)
20-38	19	Right MTM kinematics
39-41	3	PSM1 tool tip position (xyz)
42-50	9	PSM1 tool tip rotation matrix (R)
51-53	3	PSM1 tool tip linear velocity ($x'y'z'$)
54-56	3	PSM1 tool tip rotational velocity ($\alpha'\beta'\gamma'$)
57	1	PSM1 gripper angle velocity (θ)
58-76	19	PSM2 kinematics

I split the dataset into 90% training and 10% testing data

Gao, Yixin, et al. "Jhu-isi gesture and skill assessment working set (jigsaws): A surgical activity dataset for human motion modeling." *MICCAI workshop: M2cai*. Vol. 3. No. 2014. 2014.

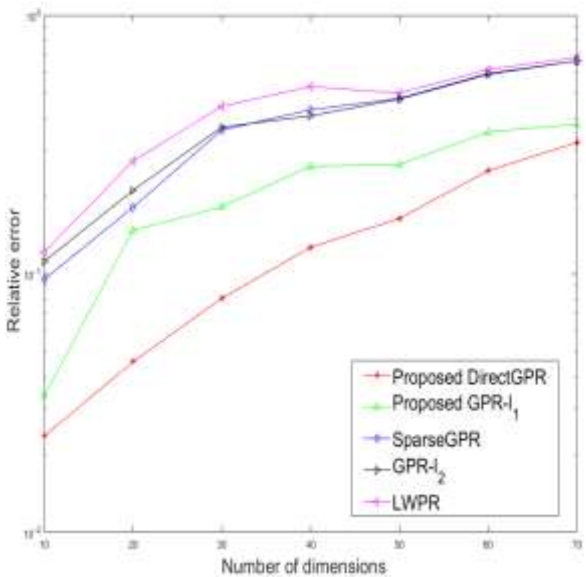
Evaluation (Knot-Tying Task)

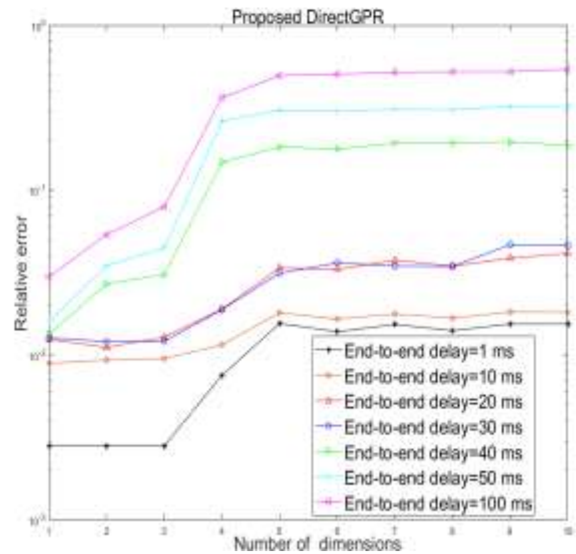


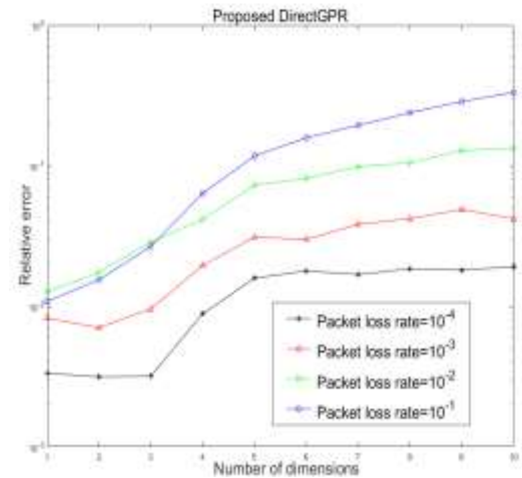


Evaluation

- We use haptic traces of different surgical tasks..







Francis Boabang, A. Ebrahimzadeh, R. Glitho, H. Elbiaze, M. Maier and F. Belqami, "**A Machine Learning Framework for Handling Delayed/Lost Packets in Tactile Internet Remote Robotic Surgery**," in IEEE Transactions on Network and Service Management, doi: 10.1109/TNSM.2021.3106577

First Contribution Conclusion

- Developed an efficient, lightweight framework for predicting haptic commands in robotic surgery
- Numerical simulations show the model balances performance and computational cost
- The proposed DirectGPR alternates between convex relaxation and nonconvex optimization to optimize a smooth, nonconvex proxy for the original cost function

- **Francis. Boabang**, Enhanced Stochastic Gradient Descent Algorithm for Machine Learning Training and Its Application in Remote Surgical Gesture Recognition, (Under Review in IEEE Transaction on Neural Network and Learning Systems)

A Review of Stochastic Gradient Descent (SGD)

- Stochastic gradient descent uses the same learning rate for all the coordinates
- Adaptive stochastic gradient descent uses different effective learning rate for different coordinates
- The idea is to propose a new adaptive learning rate to learn the varying stepsize for different coordinates
- Each coordinate has it own learning rate
- Examples of adaptive stochastic gradient descent

✓ ADAGRAD – Conventional adaptive stochastic gradient descent

✓ ADAM- Adaptive Moment Estimation [1]

✓ PADAM-Partial Adaptive Moment Estimation

[1] Kingma, Diederik P., and Jimmy Ba. "**Adam: A method for stochastic optimization**." *arXiv preprint arXiv:1412.6980* (2014).

Zaheer, Manzil, et al. "**Adaptive methods for nonconvex optimization.**" Advances in Neural Information Processing Systems. 2018.

A Review of Adaptive Stochastic Gradient Descent

- Adaptive stochastic gradient descent

Adaptive Moment Estimation(ADAM)

$$W_{t+1} = \left(W_t - \frac{\alpha_{base}}{(H_t + \epsilon)^{\frac{1}{2}}} n_t \right)$$

- With the ADAM optimizer, a small diagonal matrix H_t makes the effective learning rate $\frac{\alpha_{base}}{(H_t+\epsilon)^{\frac{1}{2}}}$ large, and large diagonal matrix H_t makes the effective learning rate $\frac{\alpha_{base}}{(H_t+\epsilon)^{\frac{1}{2}}}$ small

- **Naive Solution**: schedule to have a smaller learning rate over time to prevent certain small coordinate values from overshooting [2]

- Smaller learning rate leads to learning rate dilemma [2]

- Learning rate dilemma occurs when the optimizer fails to make an impact at the later stage of training

[2] Chen J, Zhou D, Tang Y, Yang Z, Cao Y, Gu Q. Closing the generalization gap of adaptive gradient methods in training deep neural networks. arXiv preprint arXiv:1806.06763. 2018 Jun 18.

Related Work

Summary of works related to Adaptive Stochastic Gradient Descent

Optimizer	ADAM	PADAM	Amsgrad	Wada
Large $H_{t,}$	D_{iff1}	D_{iff1}	D_{iff1}	D_{iff1}
Small H_t	D_{iff1}	D_{iff1}	D_{iff1}	D_{iff1}
$\widehat{H}_t = \max(\widehat{H}_{t-1}, H_t)$	No	Yes	Yes	No
Convergence rate for Convex Setting	$O\left(\frac{1}{\sqrt{T}}\right)$	$O\left(\frac{1}{T^{\frac{1}{2}}}\right)$	$O\left(\frac{1}{\sqrt{T}}\right)$	$O\left(\frac{1}{\sqrt{T}}\right)$
Convergence rate for Nonconvex Setting	$O\left(\frac{d^{\frac{1}{2}}}{T^{\frac{1}{2}}} + \frac{d}{T}\right)$	$O\left(\frac{d^{\frac{1}{2}}}{T^{\frac{3}{4}-\frac{s}{2}}} + \frac{d}{T}\right)$	$O\left(\frac{d^{\frac{1}{2}}}{T^{\frac{1}{2}}} + \frac{d}{T}\right)$	$O\left(\frac{lnT + d^2}{T^{\frac{1}{2}}}\right)$
Nonergodic Convergence Analysis	No	No	No	No

s : Characterize the growth of the gradient
D_{iff1} : A base learning rate
D_{iff1} :Another base learning rate

Related Work

Summary of works related to Adaptive Stochastic Gradient Descent

Optimizer	ASGD	SuperAdam	Proposed Improved Adam	Proposed Improved Amsgrad
Large $H_{t,}$	D_{iff1}	D_{iff1}	D_{iff1}	D_{iff1}
Small H_t	D_{iff1}	D_{iff1}	D_{iff2}	D_{iff2}
$\widehat{H}_t = \max(\widehat{H}_{t-1}, H_t)$	No	No	No	Yes
Convergence rate for Convex Setting	$O\left(\frac{1}{\sqrt{T}}\right)$	$O\left(\frac{1}{T^{\frac{1}{2}}}\right)$	$O\left(\frac{1}{\sqrt{T}} + \psi\right)$	$O\left(\frac{1}{\sqrt{T}} + \psi\right)$
Convergence rate for Nonconvex Setting	$O\left(\frac{lnT + d^2}{T^{\frac{1}{2}}}\right)$	$O\left(\frac{lnT + d^2}{T^{\frac{1}{2}}}\right)$	$O\left(\frac{d^{\frac{1}{2}}}{T^{\frac{1}{2}}} + \frac{d}{T} + \psi\right)$	$O\left(\frac{d^{\frac{1}{2}}}{T^{\frac{1}{2}}} + \frac{d}{T} + \psi\right)$
Nonergodic Convergence Analysis	Yes	No	Yes	Yes

ψ: Characterize the growth of the gradient
D_{iff1} : A base learning rate
D_{iff1} :Another base learning rate

Proposed Adaptive Stochastic Gradient Descent Algorithm

- We resolve the learning rate dilemma problem with Adam optimizer by selecting the base learning rate according to the size of the coordinate values H_t

- We propose a linear function

$$\alpha_{base} = u.f(H) + C$$

- For learning the coordinate values, we defined a piecewise function below

Piecewise function for base learning rate selection

$$\alpha_{base} = \begin{cases} \alpha_{min} & if\ H_t\ is\ small \\ \alpha_{max} & if\ H_t\ is\ large \end{cases}$$

Proposed Adaptive Stochastic Gradient Descent

The proposed adaptive stochastic gradient descent for small and large coordinate values is found below

Small coordinate values update rule

$$W_{t+1,small} = \left(W_t - \frac{\alpha_{min}}{(H_t + \epsilon)^{\frac{1}{2}}} n_t \right)$$

Large coordinate values update rule

$$W_{t+1,large} = \left(W_t - \frac{\alpha_{max}}{(H_t + \epsilon)^{\frac{1}{2}}} n_t \right)$$

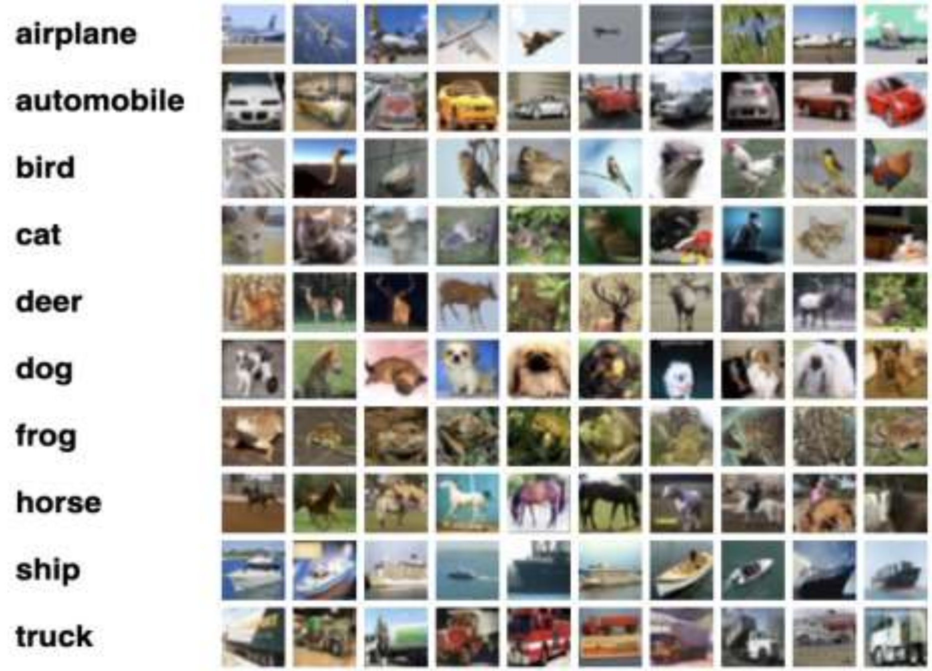
Proposed Algorithm (Improved Adam/Amsgrad)

Algorithm 1 Proposed ASGD

Input: Base learning rate $\{\alpha_{base}\}$ of the ASGD update, ten layer LSTM model, adaptive parameter β_1, β_2, preconditioner construction H, small constant $0 < \epsilon << 1$ and weight W.
Initialize: W_o, H_o, n_o
for $t = 1, ..., T$ **do**
$z_t = \nabla g(W_t)$
$n_t = \beta_1 n_{t-1} + (1 - \beta_1) z_t$
$H_t = \beta_2 H_{t-1} + (1 - \beta_2) z_t^2$
$f_{min} = \frac{sgn(\mathbb{E}[|(H_t)|] - [H_t]) + 1}{2}$
$f_{max} = \frac{sgn([H_t] - \mathbb{E}[|(H_t)|]) + 1}{2}$.
if number of nonzeros$(f_{\max}) < 2 *$ number of nonzeros$(f_{\min})$
$W_{t+1,small} = \left(W_t - \frac{\alpha_{min}}{\sqrt{(H_t + \epsilon)}} n_t \right)$
else:
$W_{t+1,large} = \left(W_t - \frac{\alpha_{max}}{\sqrt{(H_t + \epsilon)}} n_t \right)$
end
end for
Return: W_{t+1}

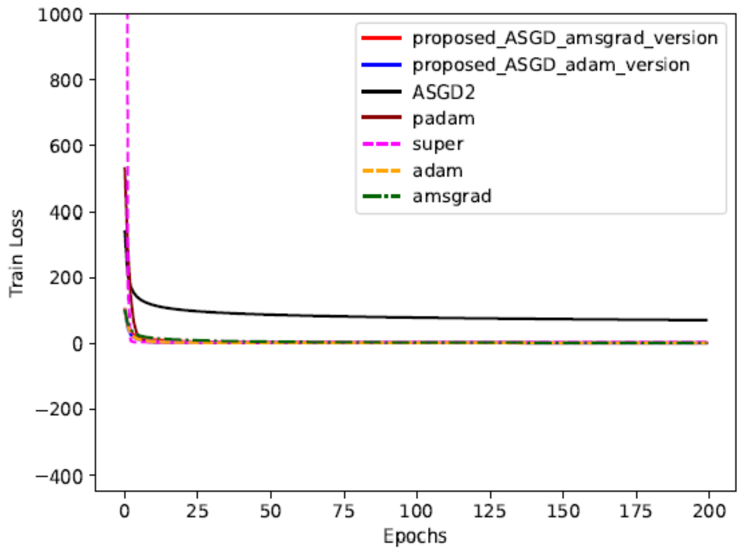
Contribution : Select large or small coordinate values update rule

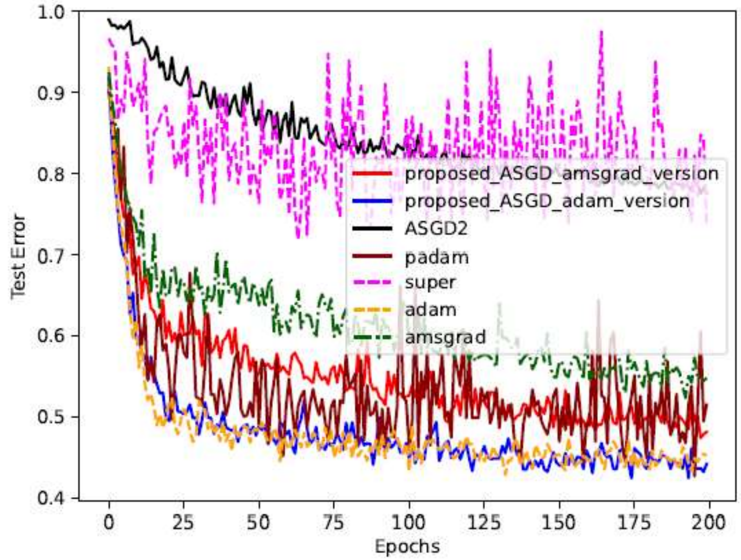
CIFAR-100 Image Dataset



- 500 training images per class
- 100 testing images per class
- 100 classes
- 20 Super classes with 5 Subclasses

Image Classification Experiment (WideResNet with Batch 64)





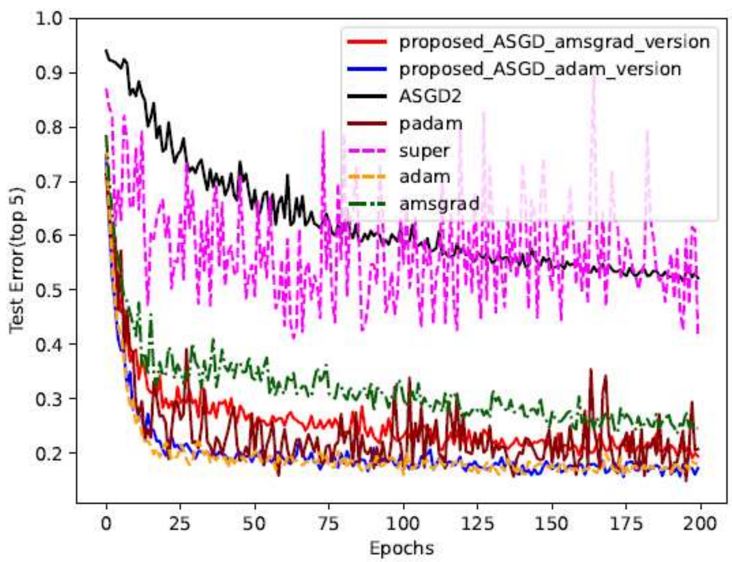
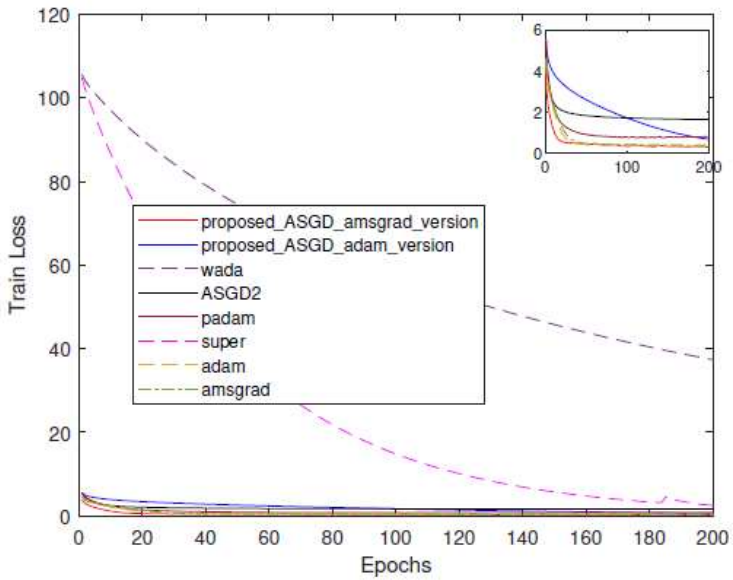
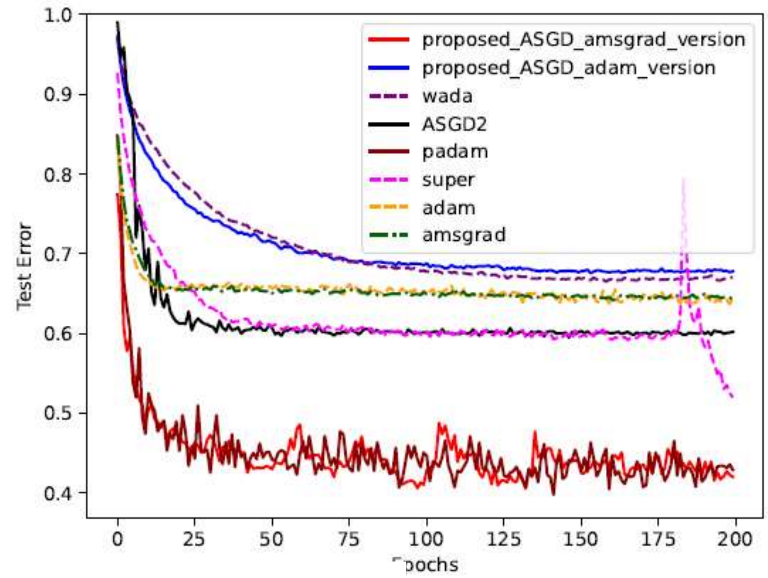
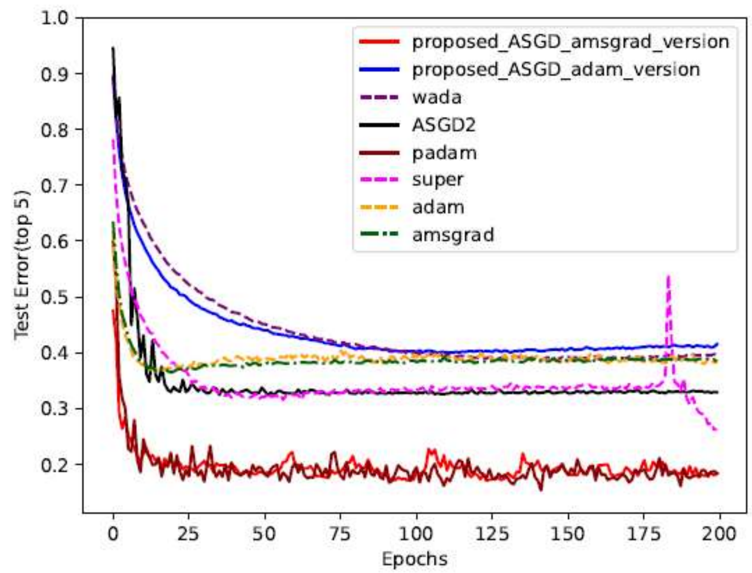


Image Classification Experiment (VGG16-Net with Batch 64)







Suturing Video Dataset

Gestures	Vocabulary
y1	Reaching for needle with right hand
y2	Positioning needle
y3	Pushing needle through tissue
y4	Transferring needle from left to right
y5	Moving to center with needle in grip
y6	Pulling suture with left hand
y8	Orienting needle
y9	Using right hand to help tighten suture
y10	Loosening more suture
y11	Dropping suture and moving to end points

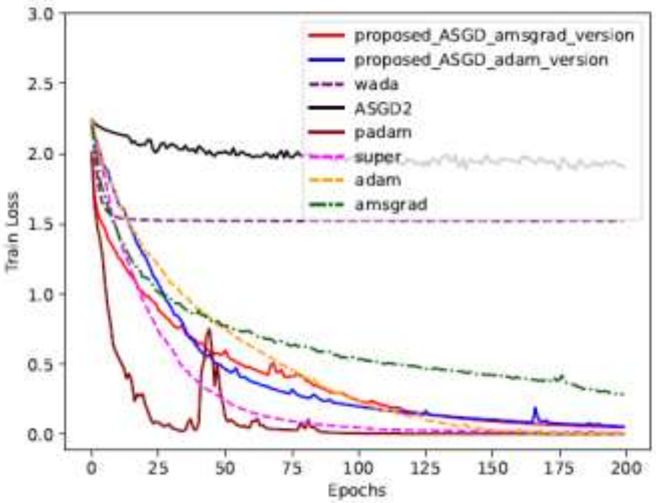
Gestures: y1, y2, y3, y4, y5, y6, y8, y9, y10,y11

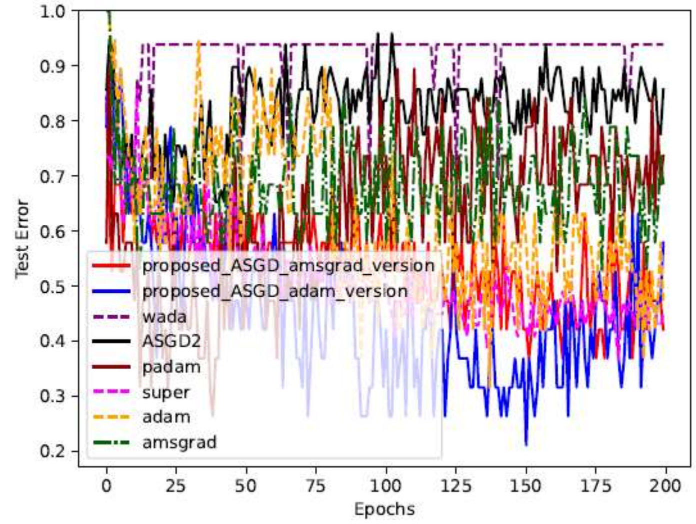
Less frequently occurring gestures: y1, y5, y8, y9, y10

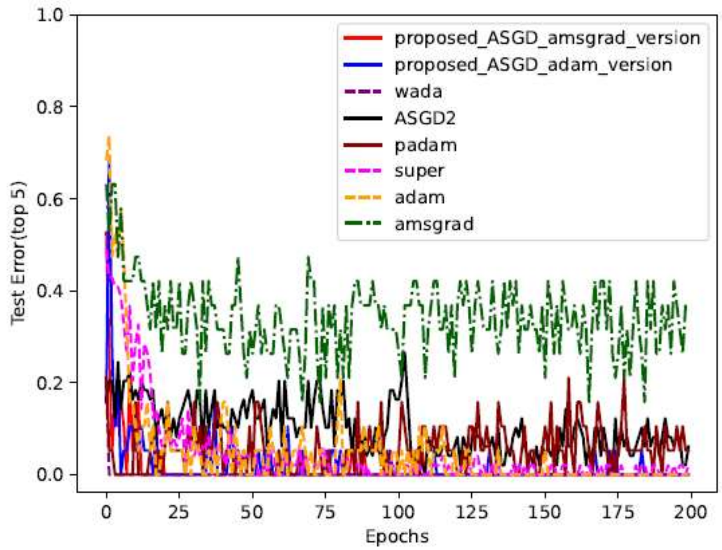
We split the dataset into 90% training and 10% testing data

Gao, Yixin, et al. "Jhu-isi gesture and skill assessment working set (jigsaws): A surgical activity dataset for human motion modeling." *MICCAI workshop: M2cai*. Vol. 3. No. 2014. 2014.

Remote Surgical Gesture Recognition (2D CNN LSTM)







Second contribution Conclusion

- For shallow neural networks, our proposed ASGD enhances generalization performance by excluding the maximum second-order momentum matrix

- In deeper neural networks, incorporating the maximum second-order momentum matrix helps counteract rapidly diminishing gradients, benefiting the optimizer

- The proposed ASGD method achieved the optimal outcome due to our approach to addressing the small learning rate dilemma problem

Conclusion of the Thesis

- The proposed approach performs best by using a convex low-rank matrix factorization as the starting point for the nonconvex method

- The proposed ASGD method achieved the optimal outcome due to our approach to addressing the small learning rate dilemma problem

- Model compression makes factorized updates to weight or covariance matrices more efficient, while ASGD is effective but costly for unfactorized updates

- Combining ASGD with model compression enables factorized updates, boosting efficiency and performance for machine learning models in tactile internet applications

Limitations

- A predefined number of iterations alternates between convex relaxed low-rank matrix factorization and nonconvex low-rank matrix factorization to scale machine learning
- The proposed ASGD algorithm cannot automatically adjust the maximum second-order momentum matrix between deep and shallow learning models, affecting remote robotic surgery performance.
- The proposed ASGD does not address the class imbalance in the JIGSAW robotic surgery dataset
- The limited availability of publicly accessible robotic surgery data for training the model has impeded our ability to pinpoint limitations in the algorithm for future enhancement

Future Work

- Combine sparse GPs with nonconvex regularization and efficient convex preserving strategies to improve predictive performance

- Expand the base learning selection strategy to other models, such as Adabrief, to assess performance in strongly convex settings

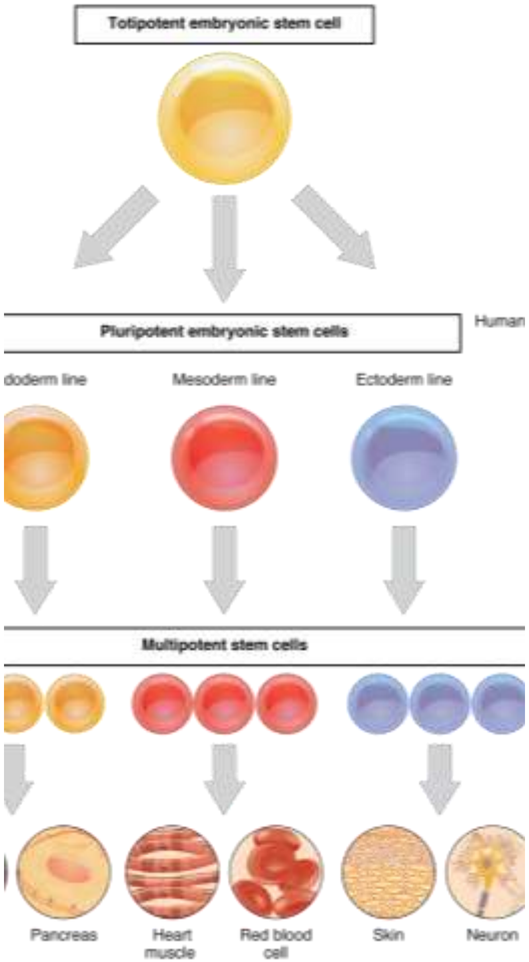
- Evaluate the proposed ASGD across various neural networks

Machine Learning Optimization Methods and Their Application to Predicting T-Cell Differentiation Using GNNs and Reinforcement Learning

Francis Boabang

Link to source codes

Stem Cell Differentiation and Machine Learning: Tunable Stem Cell Differentiation



- Stem cells are **pluripotent cells**, meaning they have the ability to differentiate into any cell type. They play a critical role in applications such as **tissue regeneration**, **cell fate determination**, **cell reprogramming** and **immunotherapies**

- In the context of **machine learning**, stem cell differentiation can be guided using **tunable parameters** such as:

✓ **Loss functions**
✓ **Constraints**
✓ **Regularization terms**
✓ **optimizers**

- These parameters help **steer the learning model** toward predicting or influencing a specific cell fate or lineage

Edgar JM, Michaels YS, Zandstra PW. Multi-objective optimization reveals time- and dose-dependent inflammatory cytokine-mediated regulation of human stem cell derived T-cell development.

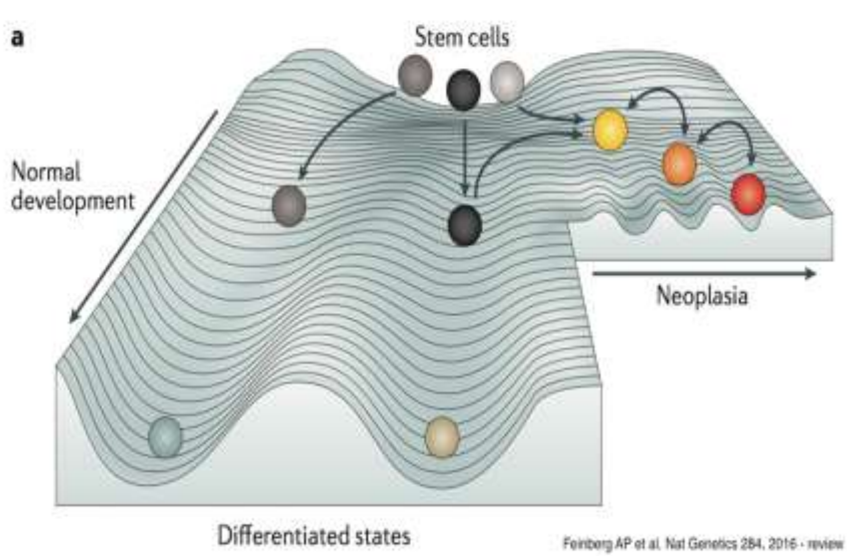
Jones, Ross D., et al. "Tunable differentiation of human CD4+ and CD8+ T cells from pluripotent stem cells." *bioRxiv* (2024): 2024-10.

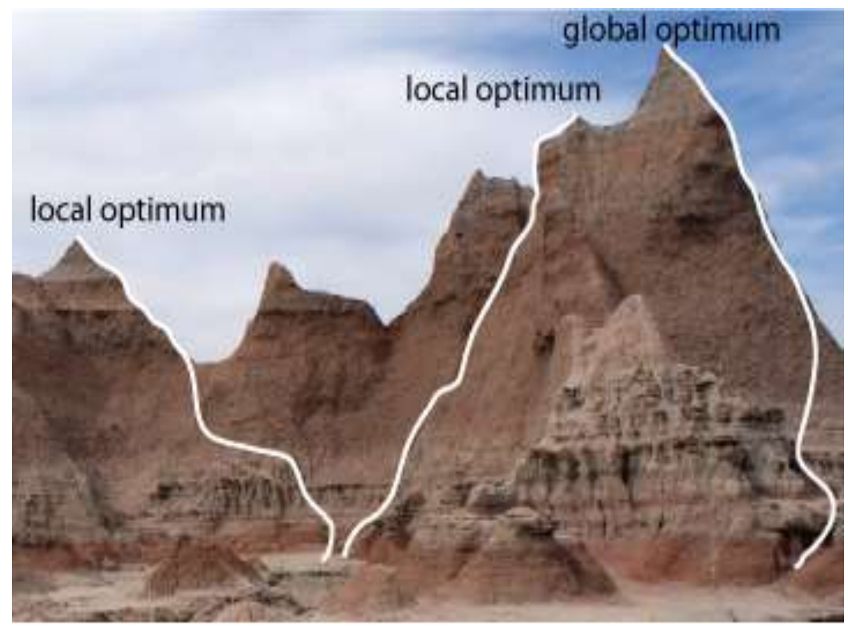
Jeffreys, Nicholas. "Amplifying mechanotransduction in human T-cell development to enhance immunotherapies." (2024).

Tuning Mechanisms in Machine Learning for Stem Cell Differentiation

- **Optimization Landscape**: Mechanical and biochemical signals form an optimization landscape, where **these forces influence stem cell fate**

- **Hyperparameter Tuning**: Fine-tuning the model's hyperparameters (e.g., learning rate, batch size, regularization strength) helps guide stem cells toward **specialized or targeted cell lineages**





IQCELL: Gene Regulatory Network and Logical Rule Inference for Predicting Effect of Gene Perturbation

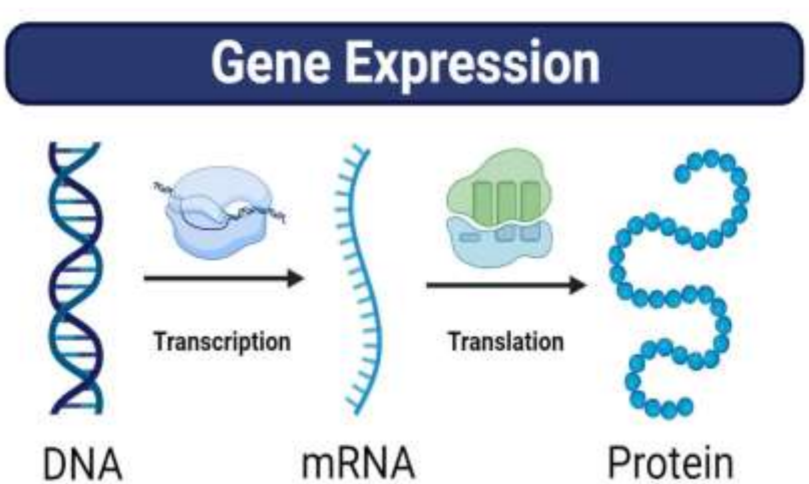
- IQCELL: predicting effect of gene perturbation on developmental trajectories
- The paper formulates gene regulatory network (GRN) as boolean logic or differential equation
- They are executable logical networks that simulate perturbed and normal cell gene expression trajectories
- Prediction or inference of gene was done via logical rule inference

Heydari, Tiam, et al. "IQCELL: A platform for predicting the effect of gene perturbations on developmental trajectories using single-cell RNA-seq data." PLoS Computational Biology 18.2 (2022): e1009907.

Recasting IQCELL as GNN + Reinforcement Learning

- We reframe GRN as graph neural network with markov decision process
- Node denote the gene state
- Action: perturbation or knockout of gene (crisper perturbation or RNAi)
- Reward desired cell fate determination

- The transition of one gene state to another was described as gene expression



Dataset:
Mouse T-cell development

Dynamic Simulation with Reinforcement Learning

- Deep Reinforcement will be used to predict the outcome of the gene perturbation

- DRL uses policy inference instead of logical rule inference for gene expression prediction.

There two types of DRL policy inference

On-policy Methods

- PPO (Proximal Policy Optimization)
- TRPO (Trust Region Policy Optimization)
- A2C (Advantage Actor-Critic)
- REINFORCE

Off-policy Methods

- TD (Temporal Difference Learning)
- SAC (Soft Actor-Critic)
- DQN (Deep Q-Networks)

Motivation for PPO Modification for Stem Cell differentiation

- **Problem:** Gene expression prediction under **knockouts/perturbations** produces **sparse datasets**

- **Challenge:** Standard PPO struggles because coordinate values can vary widely depending on perturbation thresholds

- **Goal:** Develop a more robust optimization strategy for reinforcement learning in **GRN-based simulations**

- Start from **Proximal Policy Optimization (PPO)**, a gradient-based RL optimizer.

- Modify PPO using insights from **slides 25** followed by insights from slides **14**

Standard PPO

$$L^{\mathrm{PPO}}(\theta) = \mathbb{E}_t\left[\min\left(r_t(\theta)\hat{A}_t,\ \mathrm{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t\right)\right]$$

$$r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\mathrm{old}}}(a_t|s_t)}$$

$$m_t = \beta_1 m_{t-1} + (1-\beta_1)g_t, \quad v_t = \beta_2 v_{t-1} + (1-\beta_2)g_t^2$$

$$\hat{m}_t = \frac{m_t}{1-\beta_1^t}, \quad \hat{v}_t = \frac{v_t}{1-\beta_2^t}$$

$$\theta_{k+1} = \theta_k - \eta \cdot \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon}$$

Modified PPO via the Proposed ASGDAdam/Amsgrad

$$dv_t = v_t - v_{t-1}, \quad f_{\min} = \mathbf{1}[dv_t > 0], \quad f_{\max} = \mathbf{1}[dv_t \leq 0]$$

A modification of the version in my Ph.D. thesis **slides 25** to improve differentiation of stem cell

$$\eta_t = \begin{cases} \eta_{\min}, & \text{if } \sum f_{\max} < 2 \cdot \sum f_{\min} \\ \eta_{\max}, & \text{otherwise} \end{cases}$$

$$\theta_{k+1} = \theta_k - \eta_t \cdot \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon}$$

$$L^{\text{PPOASGDAdam}}(\theta) = \mathbb{E}_t \left[\min \left(r_t(\theta)\, \hat{A}_t,\ \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\, \hat{A}_t \right) \right]$$

$$r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)}$$

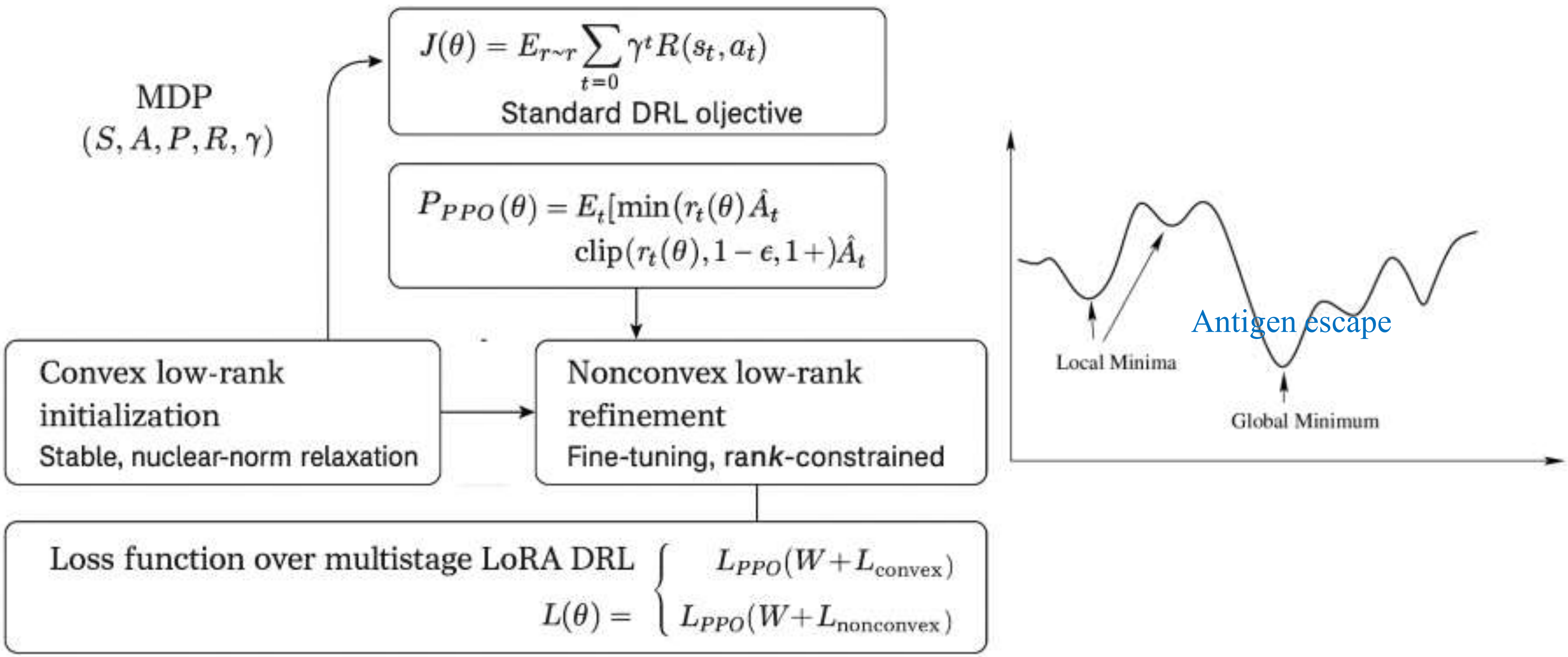
§ **Advantage:** PPOASGDADAM can handle **large/small coordinate variations** better than standard PPO.

§ **Impact:** Reduces the mean square error for **gene perturbation outcomes** in sparse single-cell datasets.

Motivation for using Low Rank Matrix Factorization for Gene Perturbation Effect Prediction

- ✓ Direct Reinforcement Learning (RL) on large Graph Neural Network (GNN)-based Gene Regulatory Networks (GRNs) is computationally prohibitive
- ✓ Scaling DRL requires **parameter-efficient strategies**
- ✓ **LoRA (Low-Rank matrix factorization):** Efficiently scale weights
- ✓ **Multistage training (convex → nonconvex):** Stabilizes learning, allowing RL to scale to larger networks without diverging taking insights from slides 11, 12 and 14

Modified PPO via the Proposed Multistage Low Rank Matrix Factorization



- Gene expression landscapes are uneven
- Naive RL may get trapped in local optima
- Multistage PPO allows:
- Convex stage : **smooth, stable exploration**
- Nonconvex stage: **fine-tuned exploitation**, capturing complex non-linear interactions

Future Work-Foundational Models for Stem Cell Differentiation

Cell embeddings (e.g., scBERT, UCE, Geneformer)

- ✓ Capture cellular states and structure
- ✓ Not directly comparable to ground-truth transcriptomic profiles (gene counts, cell fate determination, perturbation effects)

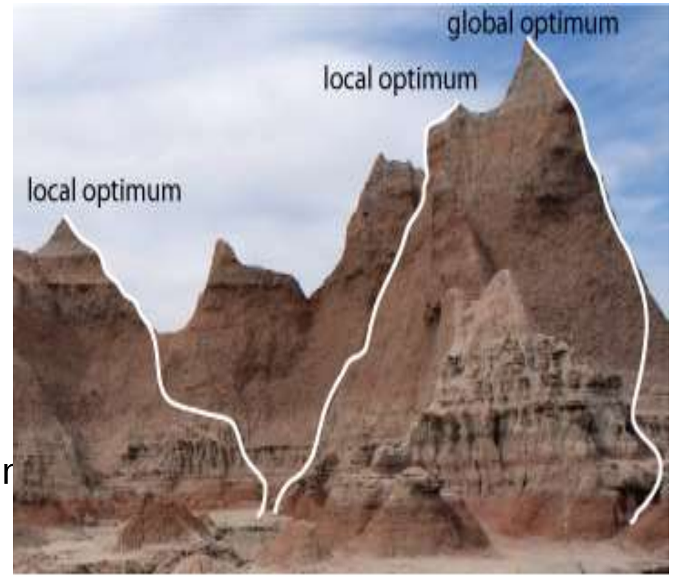
Challenge

- ✓ Need to **map embeddings to gene/protein expression values**

Method

Linear decoders:

- ✓ Multilayer perceptron (MLP)
- ✓ With or without Reinforcement Learning (RL)



Activation functions:

- ✓ Multistage convex (GReLU) & quasiconvex (ReLU) activation fur
to enhance performance
- ✓ GeLU (GReLU with fixed threshold) followed by
standard ReLU activation function

Strongly convex optimizer for improved convergence & stability

Hybrid policy optimization:

- ✓ Combine **TRPO + PPO** in a multistage setup (novel preconditioning step)
- ✓ Leverage second-order policy (TRPO) updates as preconditioning step for robustness

Thank you