Neural Networks as Model Selection with Incremental MDL Normalization

Baihan Lin

Center for Theoretical Neuroscience, Columbia University, New York, USA

Department of Applied Mathematics, University of Washington, Seattle, USA

Neuroscience inspirations

- Regularity collection in a hierarchical encoding process
  - In biological brains of primates, high-level brain areas are known to send top-down feedback connections to lower-level areas to encourage the selection of the most relevant information in the current input given the current task.

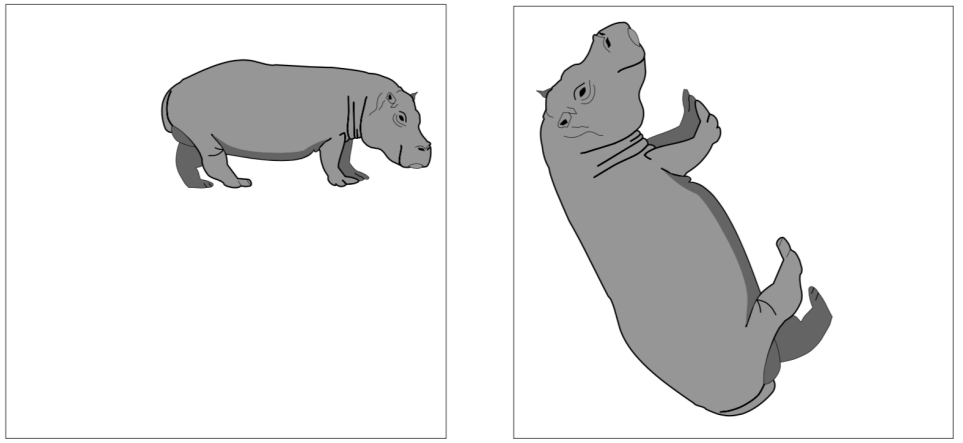
- Hierarchical neural coding in visual adaptation
  - Vertical orientation reduce their firing rates to that orientation after the adaptation.
  - While the cell responses to other orientations may increase.

- Contradicts to Bayesian assumptions (more probable, the larger firing rate)
  - However, well match the information theoretical point-of-view that the most relevant information (saliency), which depends on the statistical regularity, have higher “information”, just as the firing of the neurons.

Implicit space of neural networks

- If we consider the neural networks as population codes, the constraint space can be subdivided into the ***input-vector space***, the ***hidden-vector space***, and the ***implicit space***, which represents the underlying dimensions of variability in the other two spaces, i.e., a reduced representation of the constraint space.

- Implicit space: scale + rotation



- The relevant information about the implicit space can be constrained to ensure a minimized description length of the neural networks.

Minimum Description Length (MDL)

- Given a model class Θ consisting of a finite number of models parameterized by the parameter set θ.
- Given a data sample x, each model in the model class describes a probability P (x|θ) with the code length computed as − log P (x|θ). The minimum code length given any arbitrary θ would be given by

$$L(x|\hat{\theta}(x)) = -\log P(x|\hat{\theta}(x))$$

Compress most efficiently, yield maximum likelihood.

Minimum Description Length (MDL)

- Given a model class Θ consisting of a finite number of models parameterized by the parameter set θ.
- Given a data sample x, each model in the model class describes a probability P (x|θ) with the code length computed as − log P (x|θ). The minimum code length given any arbitrary θ would be given by

$$L(x|\hat{\theta}(x)) = -\log P(x|\hat{\theta}(x))$$

Compress most efficiently, yield maximum likelihood.

- However, the compressibility of the model, computed as the minimum code length, can be unattainable for multiple non-i.i.d. data samples as individual inputs.

optimal models vary from sample to sample~

Universal Codes

Solution:

We can define a universal code $\bar{P}(x)$ for a model class Θ, such that for any data sample x, the shortest code for x is always $\mathrm{L}\left(x\middle|\hat{\theta}(x)\right)$.

- However, the compressibility of the model, computed as the minimum code length, can be unattainable for multiple non-i.i.d. data samples as individual inputs.

optimal models vary from sample to sample~

Normalized Maximum Likelihood (NML)

- To select for a proper optimal universal code, a cautious approach would be to assume a worst-case scenario in order to make “safe” inferences about the unknown world.

$$R(p\|\Theta) = \max_q E_q[\ln \frac{f(x|\hat{\theta}_x)}{p(x)}]$$

- where the “worst” distribution q(·) is allowed to be any p(x) probability distribution.

- Among the optimal universal code, the normalized maximum likelihood (NML) probability minimizes the worst-case regret and avoids assigning an arbitrary distribution to Θ.

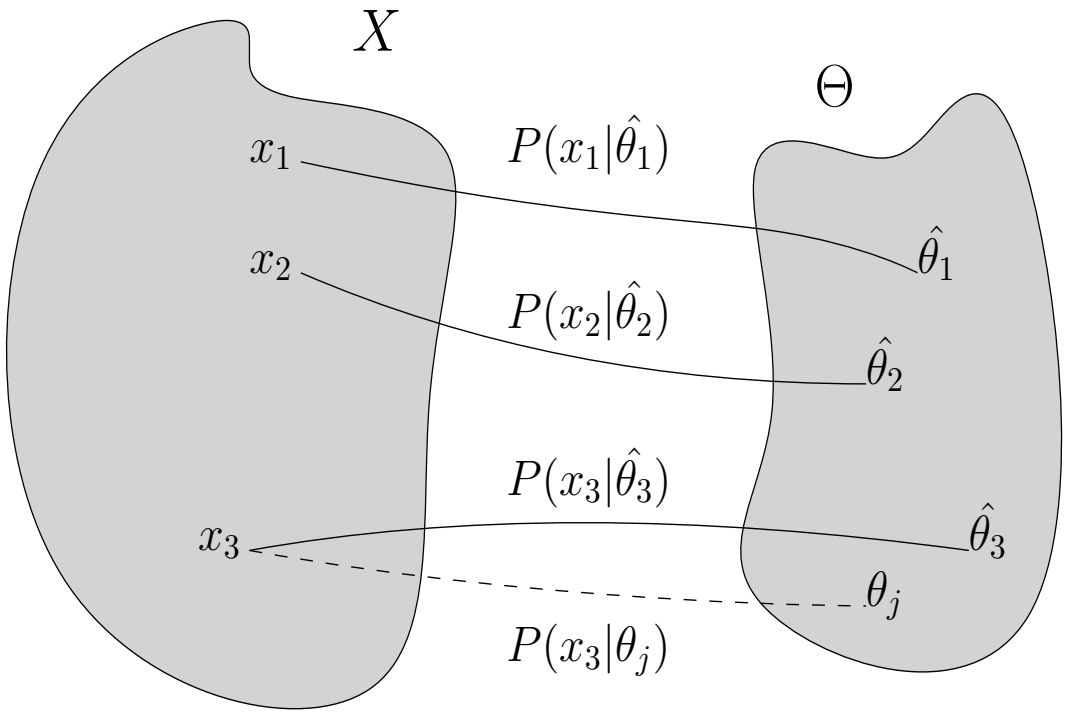
Normalized Maximum Likelihood (NML)

$$P_{NML}(x) = \frac{P(x|\hat{\theta}(x))}{\sum_{x'} P(x'|\hat{\theta}(x'))}$$

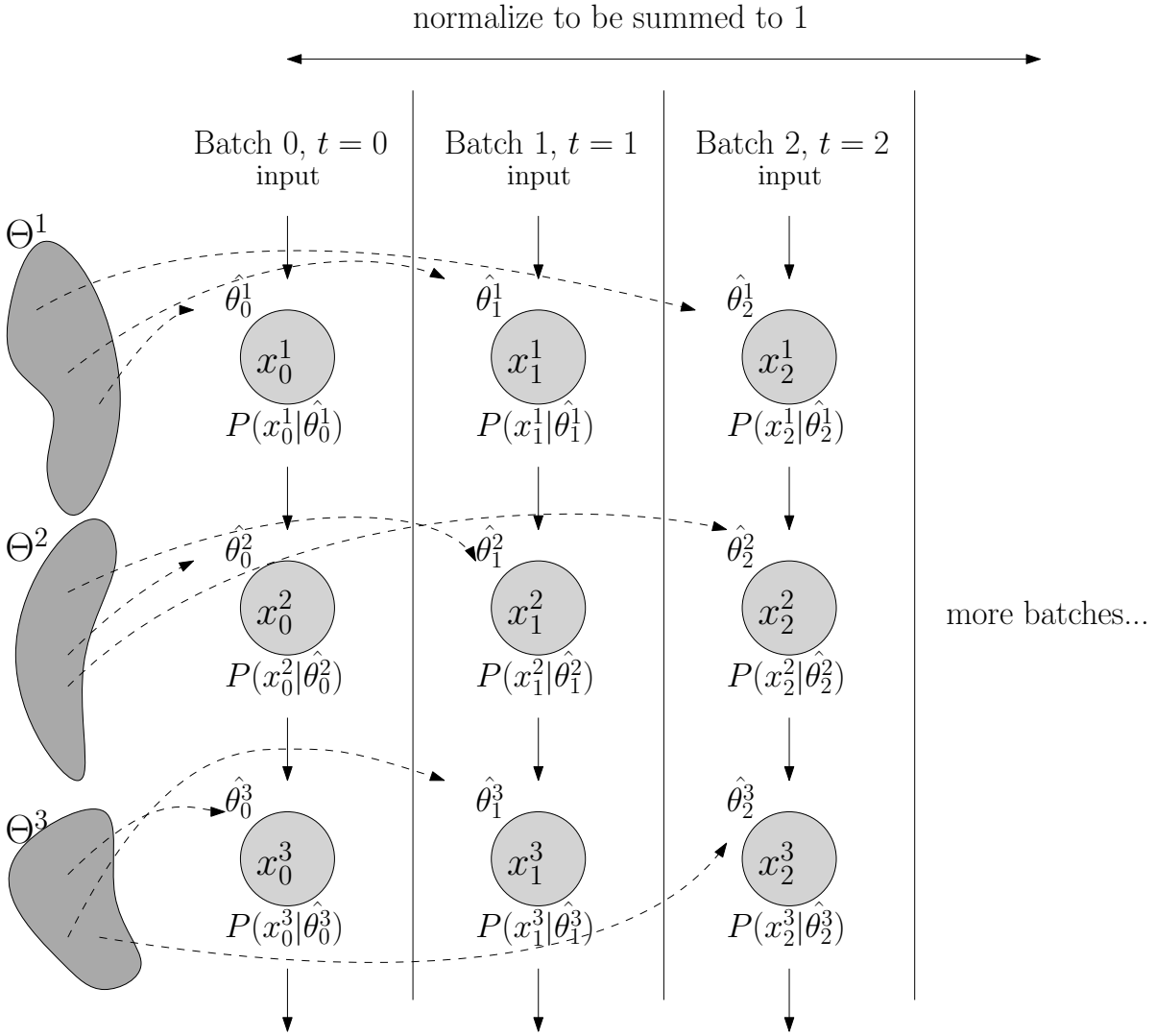
Then, the regret is the same for all data sample x is given by:

$$\begin{aligned} COMP(\Theta) &\equiv regret_{NML} \\ &\equiv -\log P_{NML}(x) + \log P(x|\hat{\theta}(x)) \\ &= \log \sum_{x'} P(x'|\hat{\theta}(x')) \end{aligned}$$

which defines the model class complexity as it indicates how many different data samples can be well explained by the model class Θ.



Neural Networks as Model Selection



$$P_{NML}(x_i) = \frac{P(x_i|\hat{\theta}_i(x_i))}{\sum_{j=0}^{i} P(x_j|\hat{\theta}_j(x_j))}$$

- where $\widehat{\theta_1}\ (x_1)$ refers to the model parameter already optimized for i − 1 steps, and have seen sequential data sample x_0 through x_{i-1}.
- This distribution is updated every time a new data sample is given, and can therefore be computed incrementally, as in batch-based training.

Regularity Normalization (RN)

Algorithm 1 Regularity Normalization (RN)

Input: Values of x over a mini-batch: $\mathcal{B} = \{x_{1,\dots,m}\}$;
Parameter: $COMP_t$, $\hat{\theta}_t$
Output: $y_i = RN(x_i)$

$$COMP_{t+1} = \text{increment}(COMP_t,\ P(x_i|\hat{\theta}_t(x_i)))$$
$$L_{x_i} = COMP_{t+1} - \log P(x_i|\hat{\theta}_t(x_i))$$
$$y_i = L_{x_i} * x_i$$

Normalization factor – optimal universal code length for the given input

Variants: SN, RLN, RBN

- Saliency Normalization (SN)
  - NML distribution can be modified to also include a data prior function, s(x)

$$P_{NML}(x) = \frac{s(x)P(x|\hat{\theta}(x))}{\sum_{x'} s(x')P(x'|\hat{\theta}(x'))}$$

  - S(x) can be the emphasis of certain inputs, to the cost of certain data, or even top-down attention etc.

- Regularity Batch Normalization (RBN)
- Regularity Layer Normalization (RLN)

Contenders for empirical evaluations

- Regularity Normalization
- Regularity Batch Normalization
- Regularity Layer Normalization
- Saliency Normalization

vs.

- Batch Normalization
- Layer Normalization
- Weight Normalization

Imbalanced MNIST problem with FFNN

- Hypothesis:
  - RN should better adapt to the regularity of the data distribution than other methods, tackling the imbalanced data issue by up-weighting the activation of the rare sample features and down-weighting those of the dominant sample features.

- To simulate changes in the context (input) distribution, in each epoch we randomly choose n classes out of the ten, and set their sampling probability to be 0.01 (only 1% of those n classes are used in the training).

- 784-1000-1000-10 FFNN with ReLU

Imbalanced MNIST problem with FFNN

Table 1. Heavy-Tailed Scenario 1: Under-represented Minorities
Test errors of the imbalanced permutation-invariant MNIST 784-1000-1000-10 task

	"Balanced"	"Rare minority"		
	$n = 0$	$n = 1$	$n = 2$	$n = 3$
baseline	4.80 ± 0.15	14.48 ± 0.28	23.74 ± 0.28	32.80 ± 0.22
BN	$\mathbf{2.77 \pm 0.05}$	12.54 ± 0.30	21.77 ± 0.25	30.75 ± 0.30
LN	3.09 ± 0.11	8.78 ± 0.84	14.22 ± 0.65	20.62 ± 1.46
WN	4.96 ± 0.11	14.51 ± 0.44	23.72 ± 0.39	32.99 ± 0.28
RN	4.91 ± 0.39	8.61 ± 0.86	14.61 ± 0.58	19.49 ± 0.45
RLN	5.01 ± 0.29	9.47 ± 1.21	$\mathbf{12.32 \pm 0.56}$	22.17 ± 0.94
LN+RN	4.59 ± 0.29	$\mathbf{8.41 \pm 1.16}$	12.46 ± 0.87	$\mathbf{17.25 \pm 1.47}$
SN	7.00 ± 0.18	12.27 ± 1.30	16.12 ± 1.39	24.91 ± 1.61

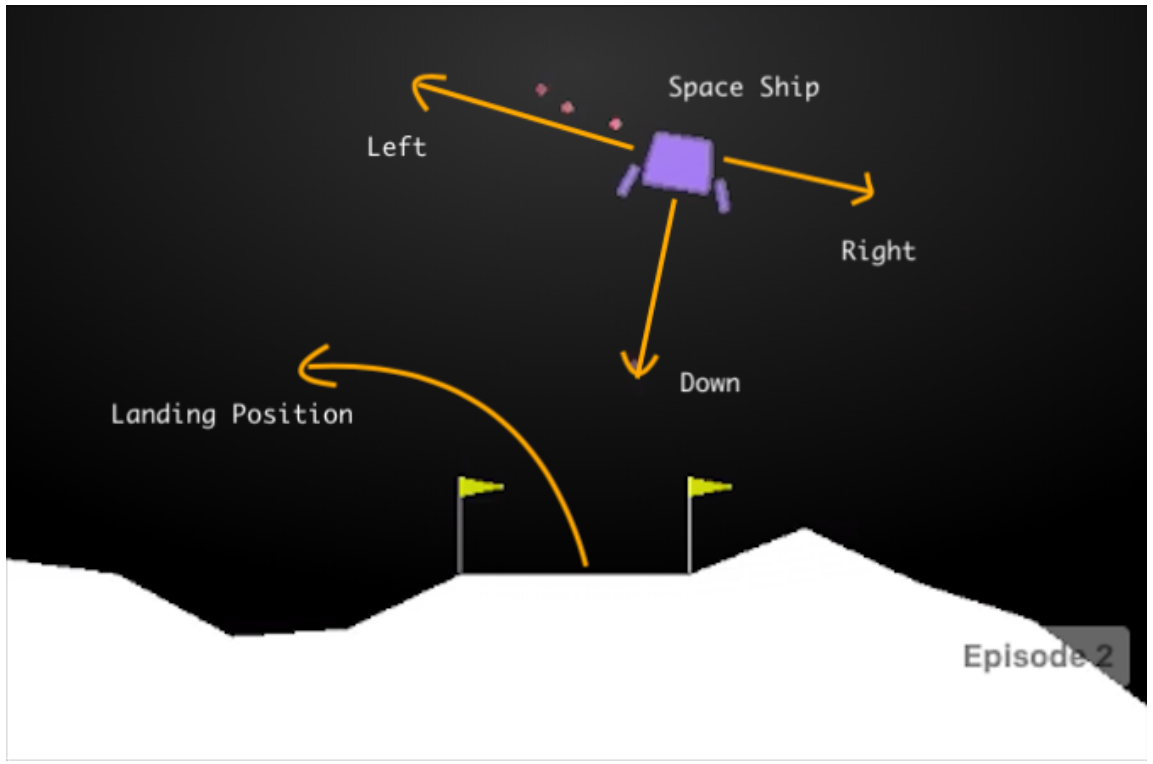
Imbalanced MNIST problem with FFNN

Table 2. Heavy-Tailed Scenario 2: Over-represented Minorities
Test errors of the imbalanced permutation-invariant MNIST 784-1000-1000-10 task

	"Highly imbalanced"				"Dominant oligarchy"	
	$n = 4$	$n = 5$	$n = 6$	$n = 7$	$n = 8$	$n = 9$
baseline	42.01 ± 0.45	51.99 ± 0.32	60.86 ± 0.19	70.81 ± 0.40	80.67 ± 0.36	90.12 ± 0.25
BN	40.67 ± 0.45	49.96 ± 0.46	59.08 ± 0.70	67.25 ± 0.54	76.55 ± 1.41	80.54 ± 2.38
LN	26.87 ± 0.97	34.23 ± 2.08	36.87 ± 0.64	41.73 ± 2.74	$\mathbf{41.20 \pm 1.13}$	$\mathbf{41.26 \pm 1.30}$
WN	41.95 ± 0.46	52.10 ± 0.30	60.97 ± 0.18	70.87 ± 0.39	80.76 ± 0.36	90.12 ± 0.25
RN	$\mathbf{23.35 \pm 1.22}$	33.84 ± 1.69	41.47 ± 1.91	60.46 ± 2.88	81.96 ± 0.59	90.11 ± 0.24
RLN	23.76 ± 1.56	32.23 ± 1.66	43.06 ± 3.56	57.30 ± 6.33	88.36 ± 1.77	89.55 ± 0.32
LN+RN	25.65 ± 1.91	$\mathbf{28.71 \pm 1.97}$	$\mathbf{33.14 \pm 2.49}$	$\mathbf{36.08 \pm 2.09}$	44.54 ± 1.74	82.29 ± 4.44
SN	31.07 ± 1.41	41.87 ± 1.78	52.88 ± 2.09	68.44 ± 1.42	83.34 ± 1.85	82.41 ± 2.30

RL problem with DQN

- OpenAI Lunarlander-V2 environment



- DQN with two hidden units of 64 neurons
- Experience replay
- Adam optimizer

RL problem with DQN

Algorithm 1 Deep Q-learning with Experience Replay

Initialize replay memory $\mathcal{D}$ to capacity N
Initialize action-value function Q with random weights
for episode $= 1, M$ **do**
  Initialise sequence $s_1 = \{x_1\}$ and preprocessed sequenced $\phi_1 = \phi(s_1)$
  for $t = 1, T$ **do**
    With probability ϵ select a random action a_t
    otherwise select $a_t = \max_a Q^*(\phi(s_t), a; \theta)$
    Execute action a_t in emulator and observe reward r_t and image x_{t+1}
    Set $s_{t+1} = s_t, a_t, x_{t+1}$ and preprocess $\phi_{t+1} = \phi(s_{t+1})$
    Store transition $(\phi_t, a_t, r_t, \phi_{t+1})$ in $\mathcal{D}$
    Sample random minibatch of transitions $(\phi_j, a_j, r_j, \phi_{j+1})$ from $\mathcal{D}$
    Set $y_j = \begin{cases} r_j & \text{for terminal } \phi_{j+1} \\ r_j + \gamma \max_{a'} Q(\phi_{j+1}, a'; \theta) & \text{for non-terminal } \phi_{j+1} \end{cases}$
    Perform a gradient descent step on $(y_j - Q(\phi_j, a_j; \theta))^2$ according to equation 3
  end for
end for

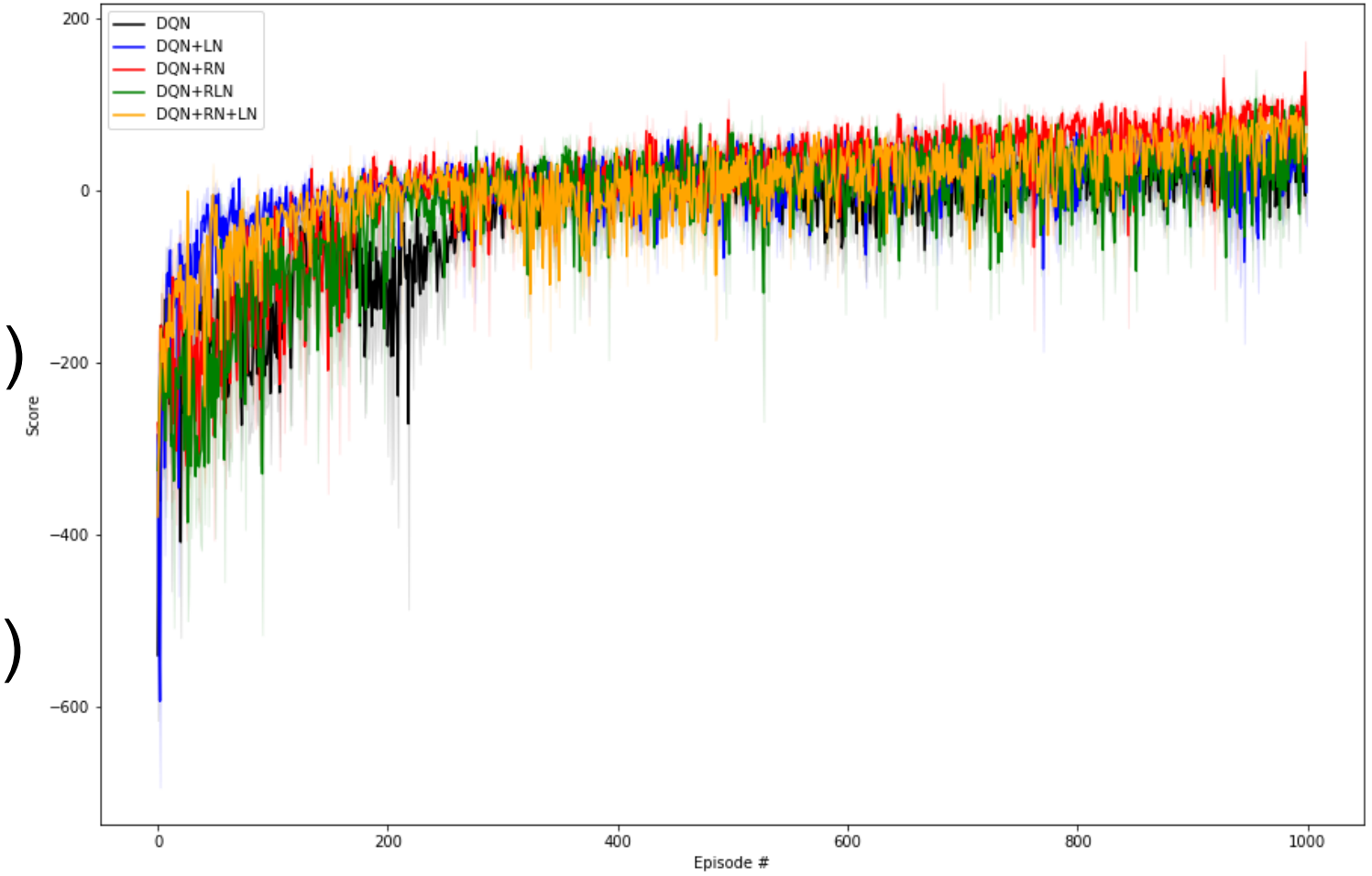
Apply RN, RLN, LN, BN, SN.
Update regularity statstics.

RL problem with DQN

- Average final scores:

- RN (76.95 ± 4.44)
- RN+LN (65.82 ± 10.91)
- RLN (49.27 ± 40.35)

- Baseline (37.17 ± 8.82)
- LN (-1.54 ± 39.14)

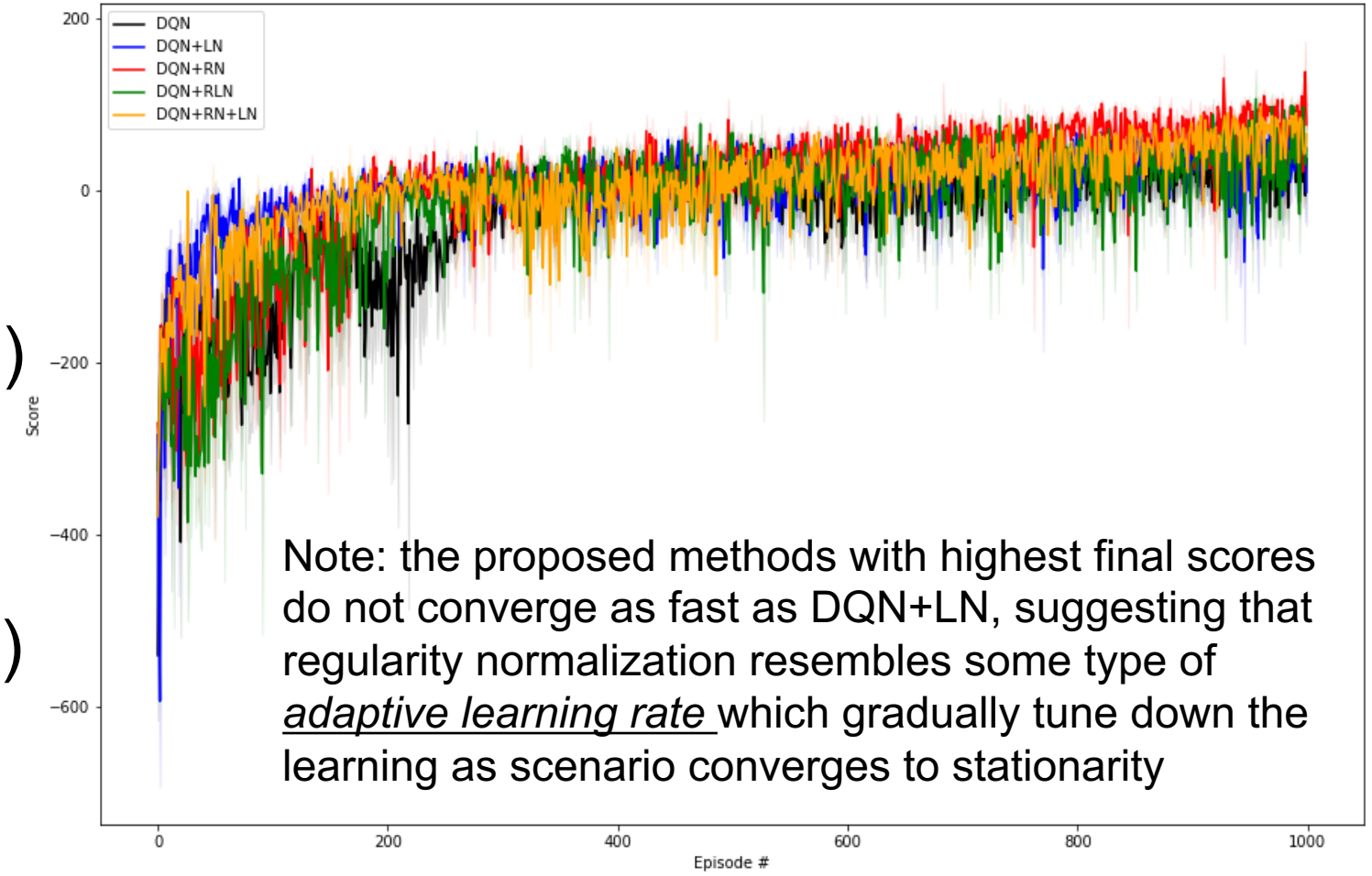


RL problem with DQN

- Average final scores:

- RN (76.95 ± 4.44)
- RN+LN (65.82 ± 10.91)
- RLN (49.27 ± 40.35)

- Baseline (37.17 ± 8.82)
- LN (-1.54 ± 39.14)



Note: the proposed methods with highest final scores do not converge as fast as DQN+LN, suggesting that regularity normalization resembles some type of *adaptive learning rate* which gradually tune down the learning as scenario converges to stationarity

Discussion and ongoing directions

- Extend to fully Bayesian variational approach to include the regularity estimation as part of the optimization process.

- Investigate the changes in overall loss or probability as the computation of NML makes selection on the model.

- MDL → ensemble modeling process and usually involves multiple models.
- Assumption
  - that the only model trained at each step is the local “best” model learned so far, but local maximal likelihood may not be a global best approach for model combinations.
- Can we theoretically support the assumption above for our greedy approach for “optimal parameters”?

Future work and next steps

- Top-down attention given by data prior (such as feature extracted from signal processing, or task-dependent information) → Top-down meta learning:
  - How different functions of s(x) behave in different task settings.

- RN as Unsupervised attention mechanism
  - Unsupverised + Supervised Attention with top-down attention from either oracle supervision or other meta information.

- Convolutional and recurrent cases.
- Non-stationary rewards in a changing environments.

Acknowledgements

Collaborators/Co-authors:

Dr. Niko Kriegeskorte (CU)
Dr. Ning Qian (CU)
Dr. Raul Rabadan (CU)
Dr. Guillermo Cecchi (IBM)
Dr. Irina Rish (IBM)
Dr. Djallel Bouneffouf (IBM)
Dr. Jenna Reiner (IBM)
Dr. Mar Gonzalez-Franco (Microsoft)
Dr. Shwetak Patel (UW)
Dr. David Baker (UW)
Dr. Hong Qian (UW)
Dr. Jaime Olavarria (UW)
Dr. Yue Teng (BIME)
Dr. Tim Kietzmann (Cambridge)

Thanks! Questions?

- Feel free to also check out my other project in IJCAI poster session:

Split Q Learning: Reinforcement Learning with Two-Stream Rewards

Poster #908 in **Hall A** August 15th
Talk in area **2605** August 12th at **13:35**