LinkedIn

Carnegie Mellon University

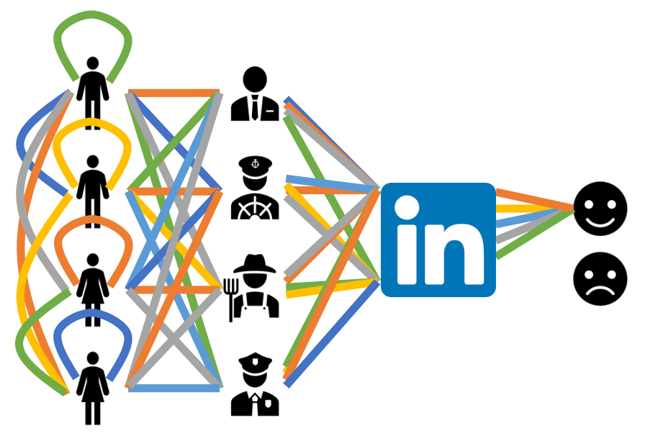
Scalable Automatic Machine Learning

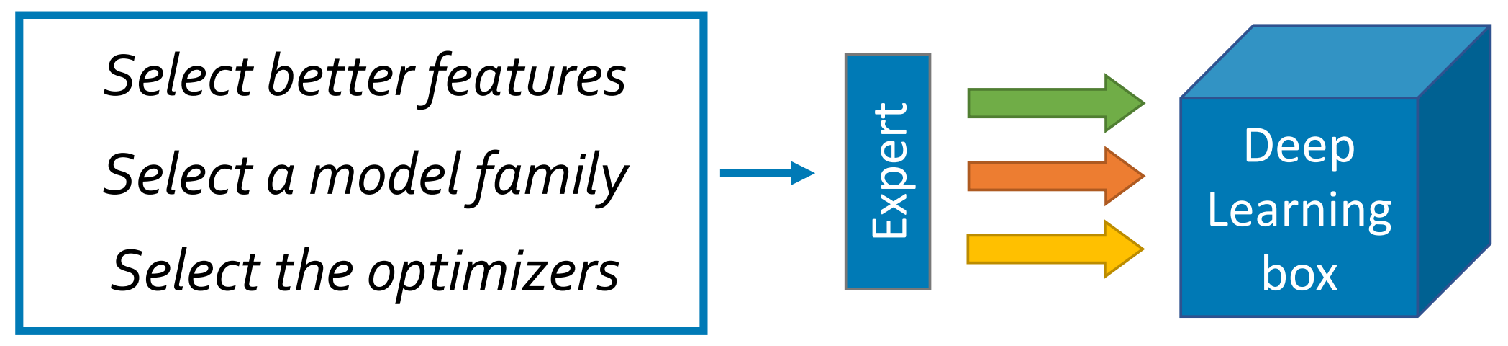
Yuwei Qiu, Chengming Jiang, Huiji Gao, Bee-Chung Chen
AI Algorithms Foundation team

LinkedIntern
Class of 2019
Summer
Sunnyvale

MOTIVATION

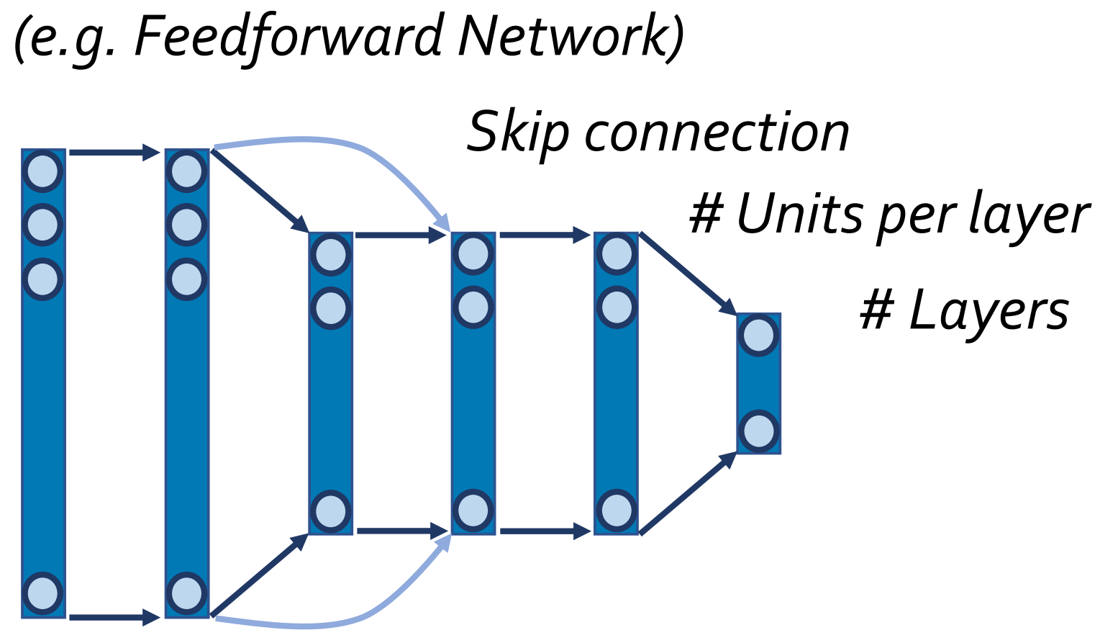
- @LinkedIn,
  - Data come and go
  - Tasks come and go
  - but learning is forever
- Learn more effectively, less trial-and-error
- Problems in so-called "end-to-end Deep Learning":
  - Great consumption of (expertise) human power
  - High cost of hyperparameter tuning/ structure learning





Select better features
Select a model family
Select the optimizers

- Performance is very sensitive to many parameters



(e.g. Feedforward Network)

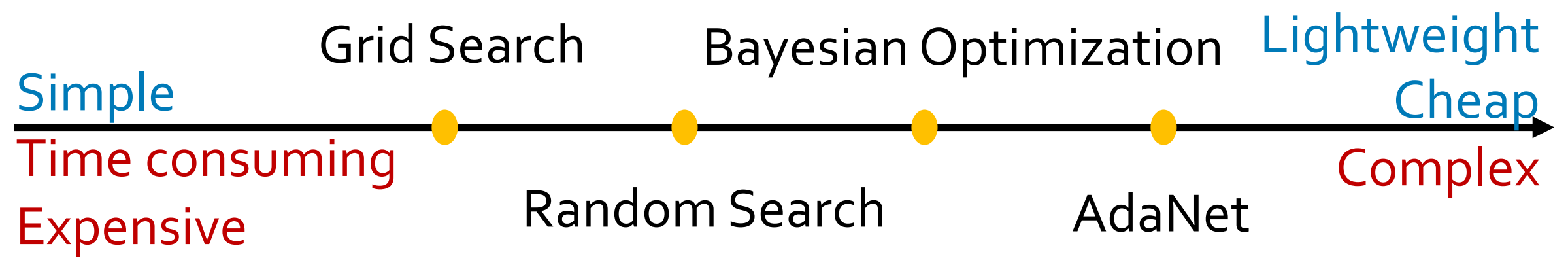
Optimization,
Learning rates,
Momentum,
Batch sizes,
Dropout rates,
Weight decay,
Activation,
Regularization
…

AUTO TUNING

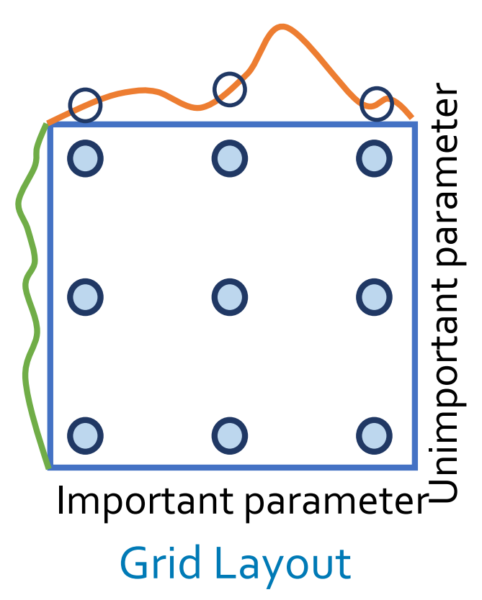
Current:
Solution =
Expertise + Data + Hours

We can turn this into:
Solution =
Data + N × Hours

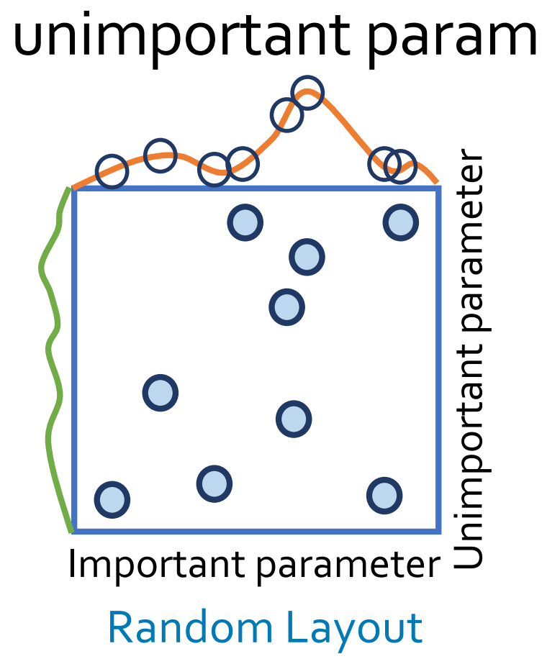
BASELINE



- Grid Search and Random Search
  - Shoot all structures with all hyperparameters
  - Then find the best set of parameters
  - Random search handles unimportant params better



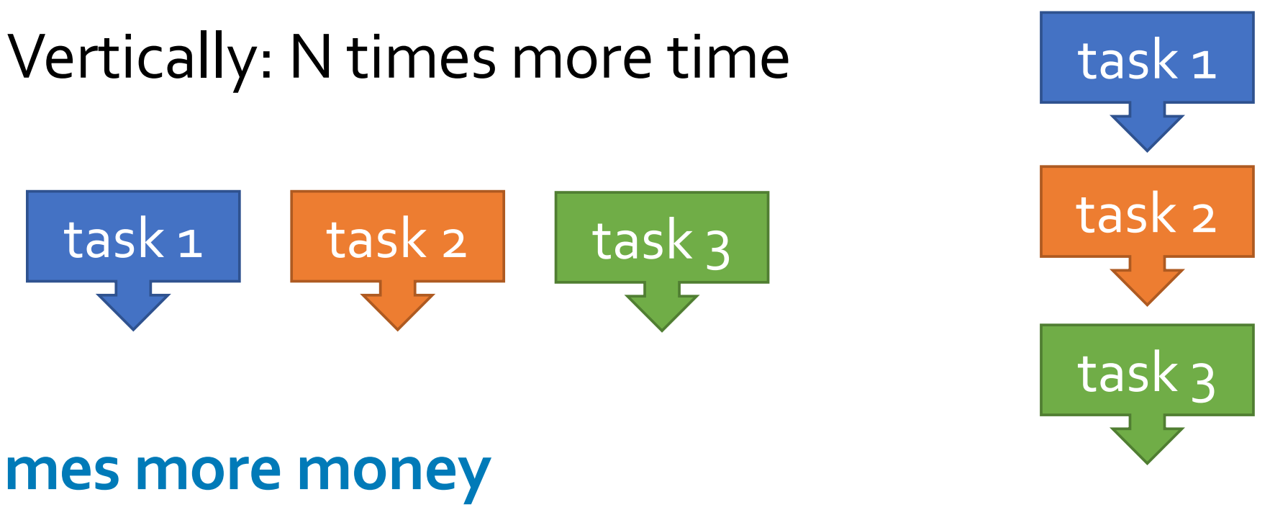
Grid Layout



Random Layout

OBSERVATION

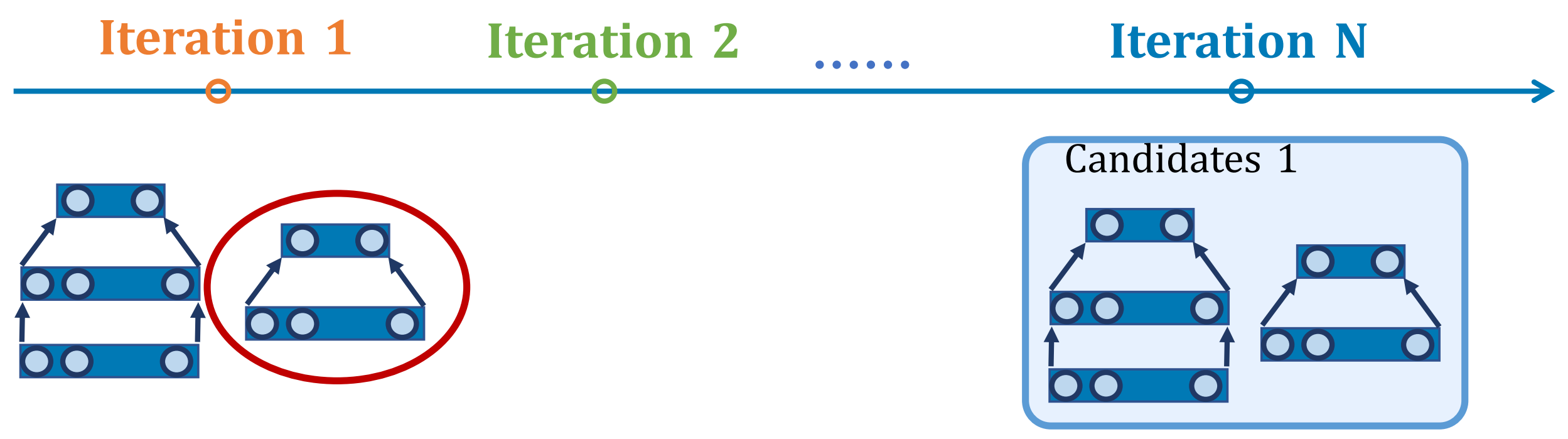
- Disastrous heavyweight structure exploration
- To grid search or random search on N set of parameters…
  - Horizontally: N times more resources
  - Vertically: N times more time



- N times more money

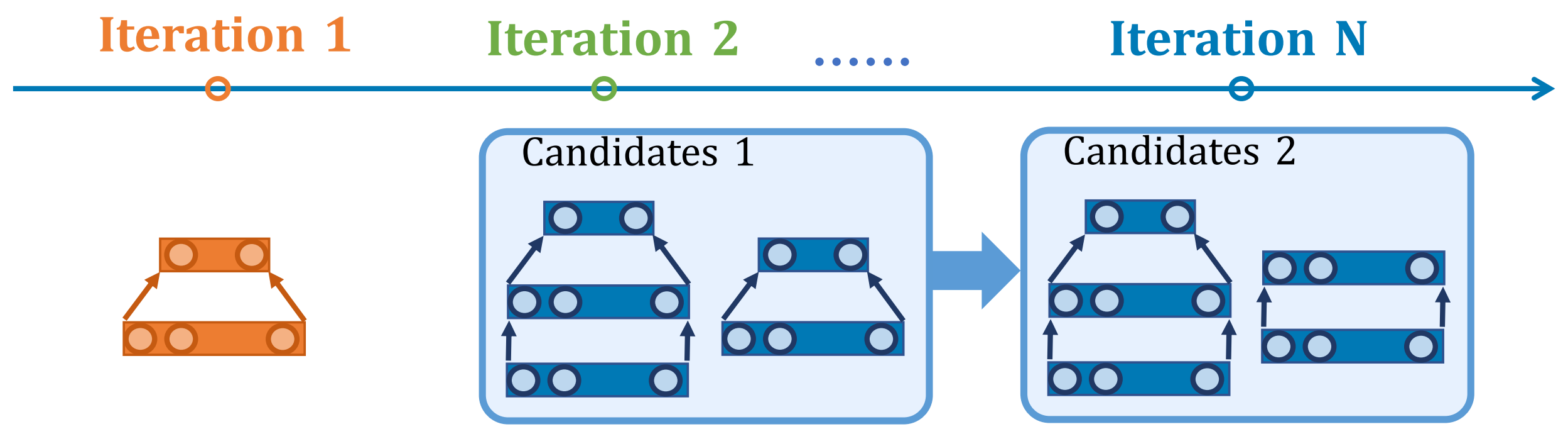
ALGORITHM

- **AdaNet: Structure grows with lightweight subnetworks.**



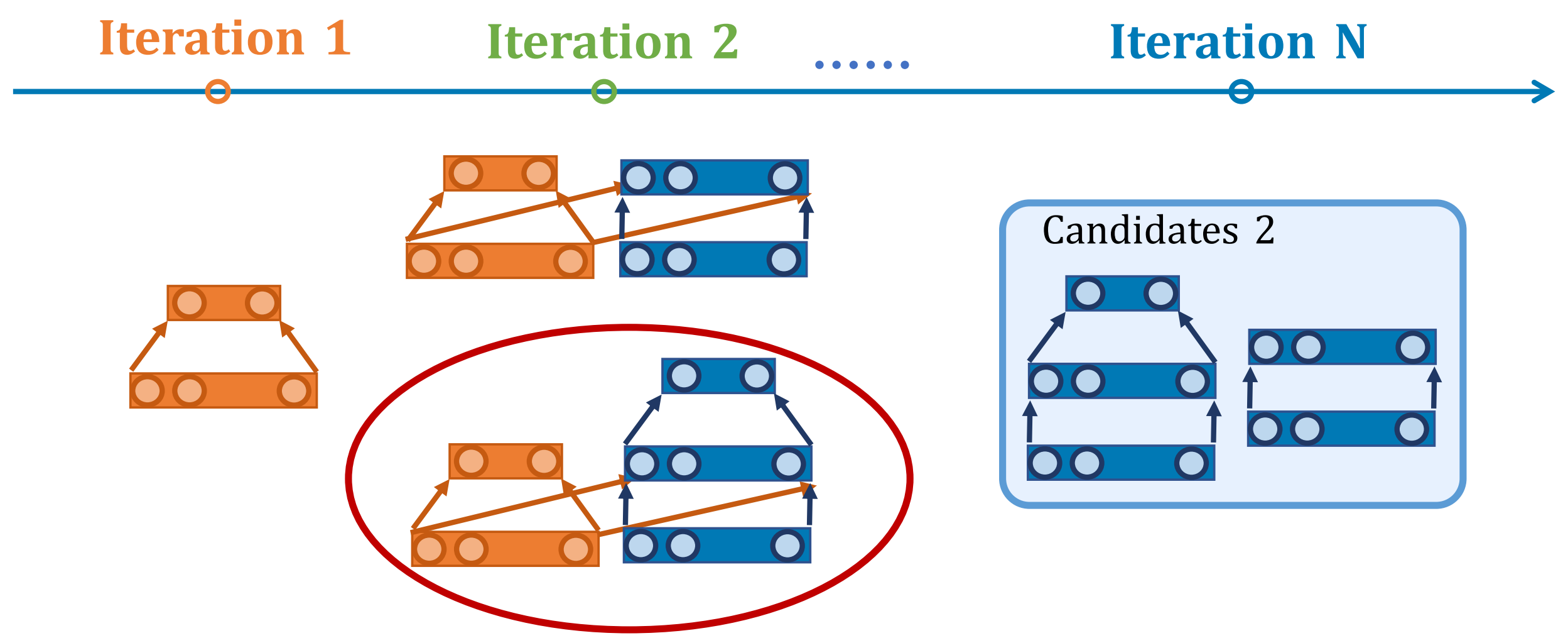
Start with random small subnetworks and train them separately

Choose the one leading to the best reduction of evaluation metric function.



Based on the evaluation results of these subnetwork, expand candidates pool under specific rules.

Train and re-train the subnetworks in the expanded candidates pool.



Augment candidates with the current network.

Then add and train new connections between subnetworks.

Choose the one leading to the best reduction of evaluation metric function.

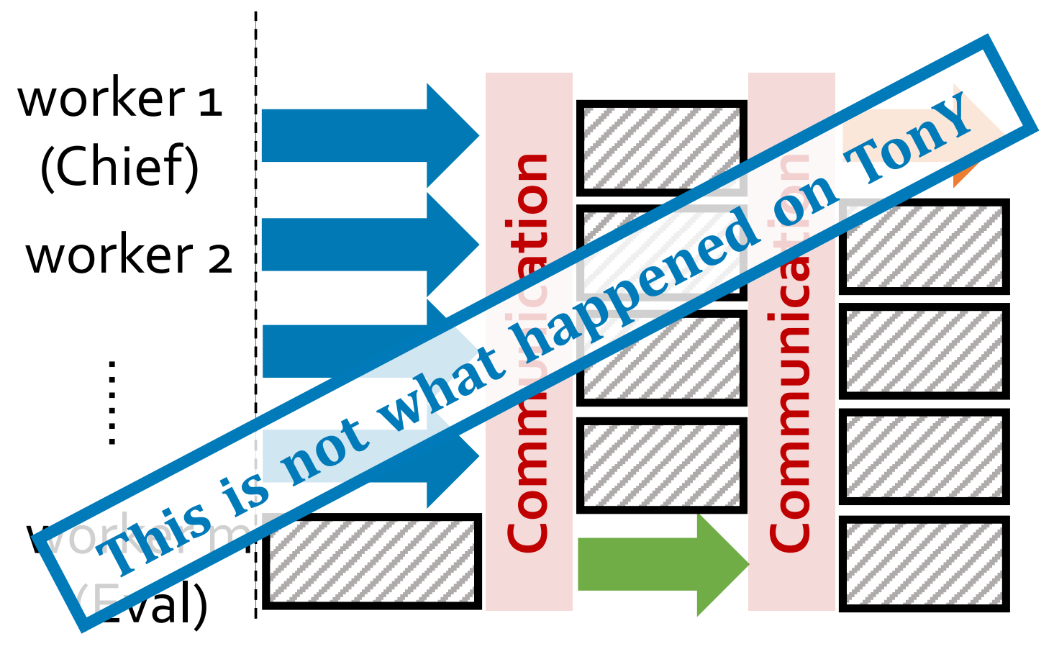
CHALLENGE ON THE CLOUD

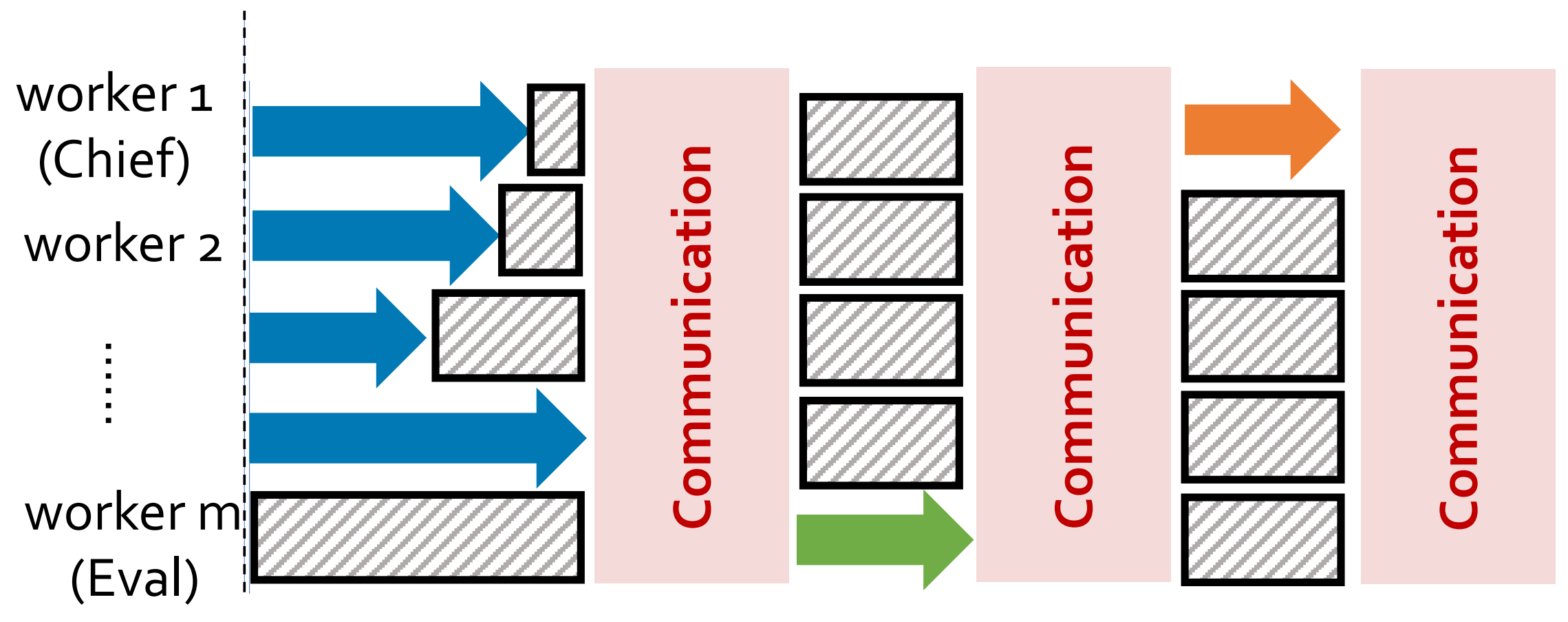
- Straggler effects, communication latency and resource occupied significantly slow down the distributed training process.

Model Tuning

Chief receives the metrics from all workers and decides the next set of parameters.

Other workers wait for Chief to finish.



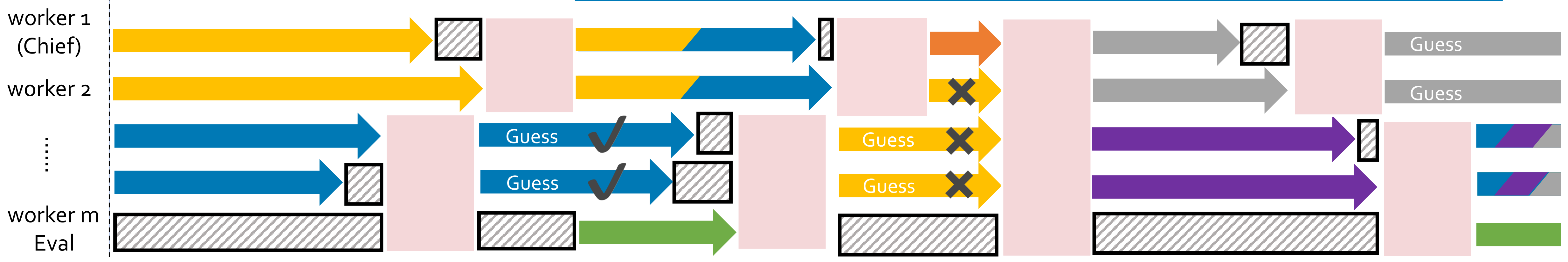


ASYNCHRONOUS MODEL TRAINING

- Observations in Sync model training:
  - Many workers involved in communication during training of subnetworks: Heavier straggler effect
  - Long waiting time during evaluation and model adjusting: Waste of resources

Asynchronous distributed model training

Subnetworks of current iteration finish, start ensemble using remaining available resources. Previously occupied nodes continued on their previous work.



Start training subnetworks separately.
Each subnetwork occupies one share of all resources.

Once finished, start to re-train of current subnetwork
Assume this subnetwork will remain in the pool. (Forecasting)

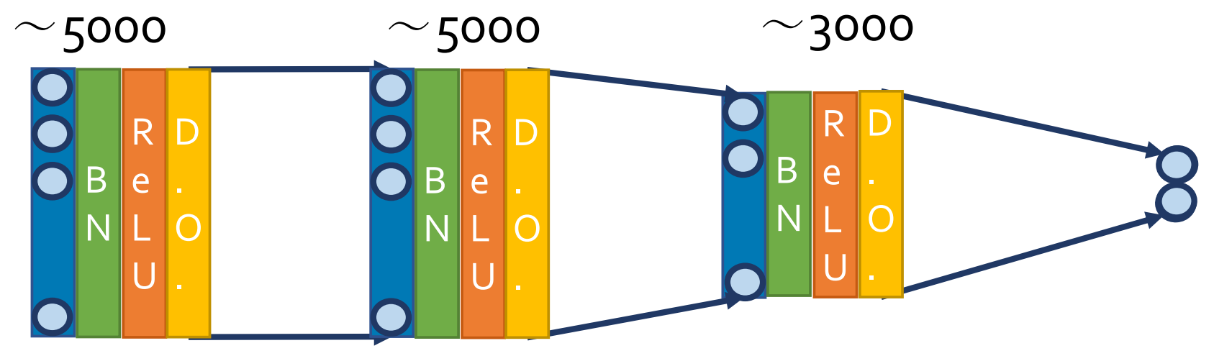
After evaluation, Chief decides subnetworks for next iteration.
- **Keep luck guess**
- **Throw away bad guess**

RESULTS

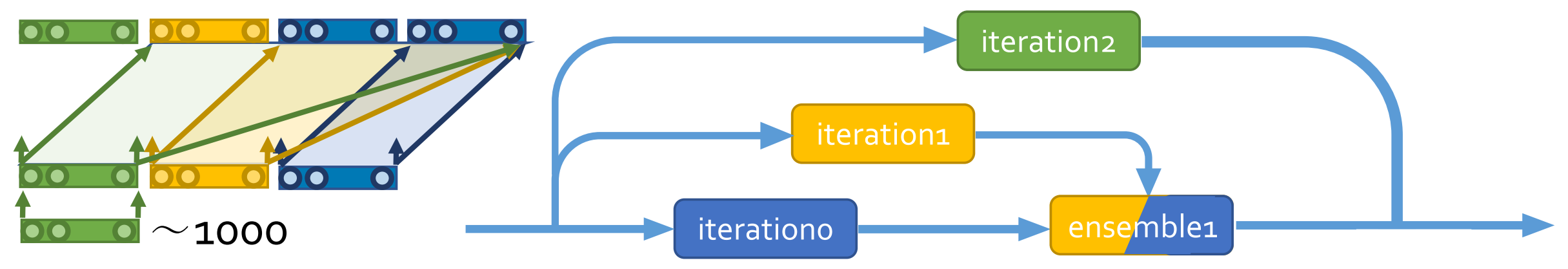
Model	Training Time	Eval Time	AUC
Logistic Regression	1 hour 45 mins	**20 mins**	66.3%
Manual Tuned	2 hours 05 mins	30 mins	66.8%
Random_Bst_NN	2 hours 35 mins	35 mins	67.2%
AdaNet	**36 mins**	31 mins	**67.6%**

We achieved **a 3 × speed-up** and **a lift of AUC of 1.3%.**

Manual Tuned Structure



AdaNet Structure



CONTRIBUTION

- Exploration of various auto tuning algorithms sheds light on future direction for different DNN model at LinkedIn.
- Wiped out disastrous heavyweight structure exploration problem in real-world auto-tuning services by introducing adaptive structure learning algorithm.
- Implemented asynchronous model training to avoid straggler effects and communication latency.

Most importantly, make everything scalable!

Contact

Yuwei Qiu
yqiu@linkedin.com

Chengming Jiang
cjiang@linkedin.com