Neural Architecture Search with Reinforcement Learning

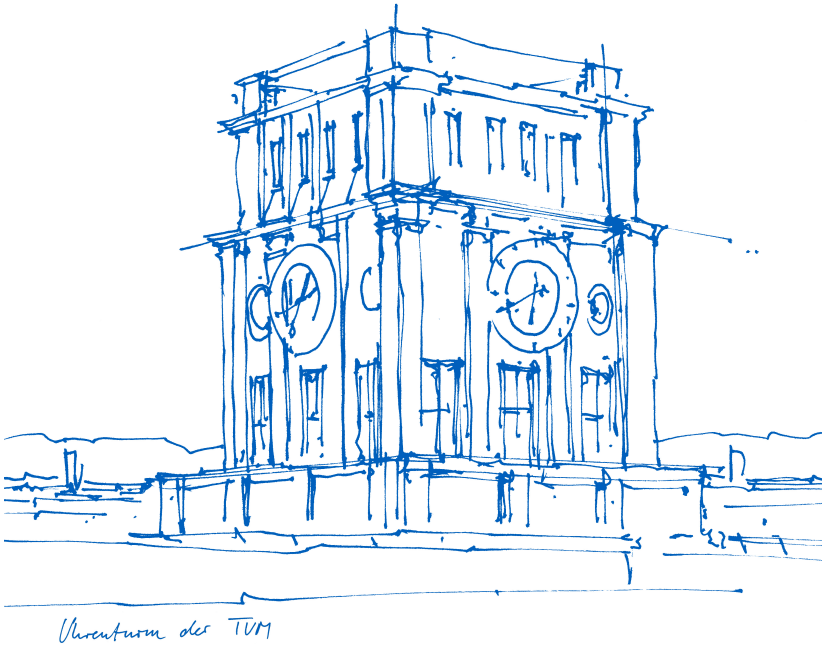
Nick Harmening

Technical University Munich

Faculty for Computer Science

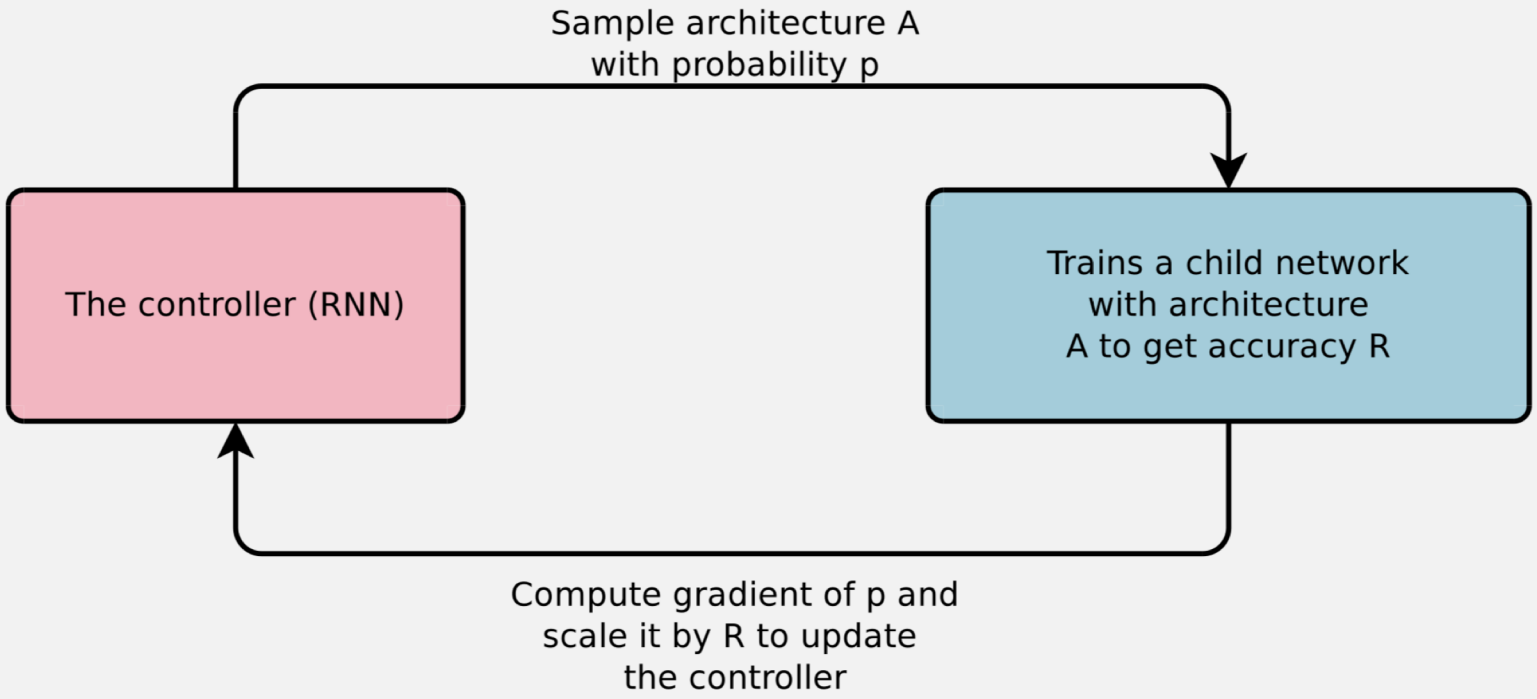
Recent trends in Automated Machine Learning

06. June 2019



Neural Architecture Search with Reinforcement Learning - Principle

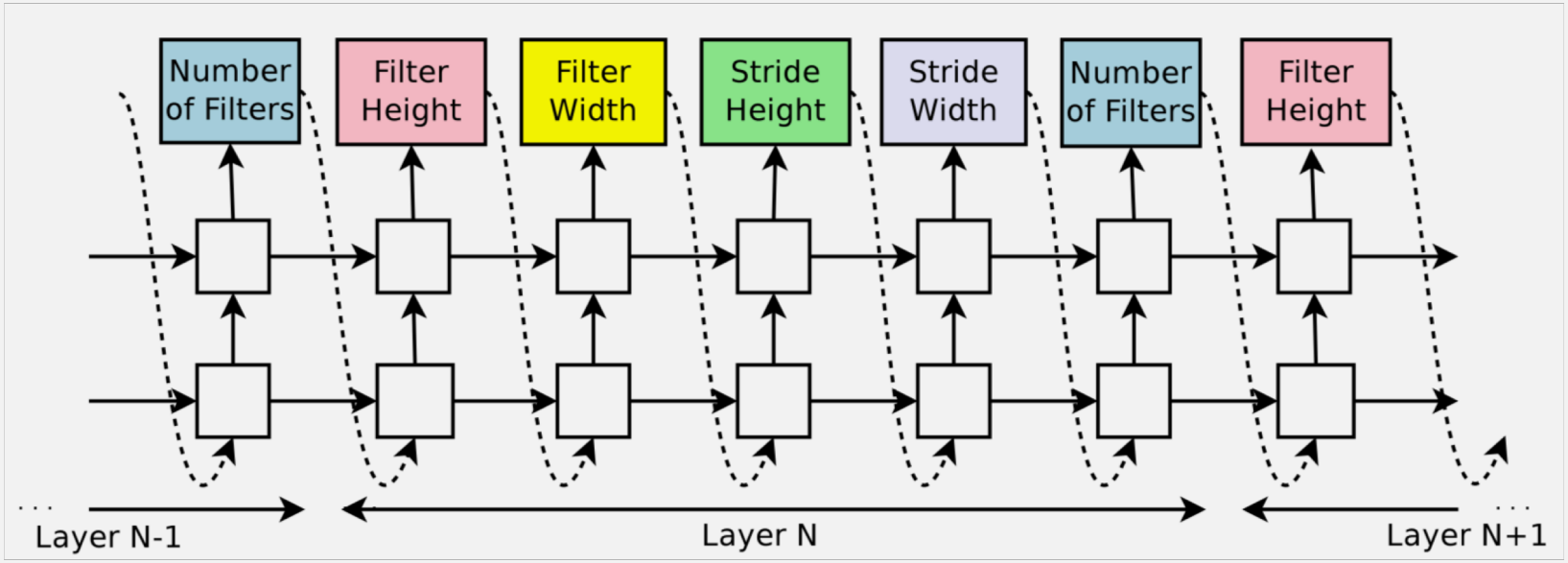
→ Automatization of CNN and RNN architecture search by a controller RNN



Finding a CNN Architecture with a RNN Controller

Sampling a CNN Child Network

- CNN architecture can be modelled as a list of tokens
- The controller RNN samples tokens for a fixed number of layers
- The number of layers is increased throughout the controllers training progress



Training the Controller with REINFORCE

1. Sample a child network
2. Train the sampled child network until convergence
3. Optimize parameters θ_c of the Controller RNN to maximize the validation accuracy R of the child network

Objective Function for Policy Gradient: $\mathcal{L}_{\theta_c} = \mathbb{E}[R_t],\ R_t = \sum_{k=t}^{T-1} \gamma^{k-t} r_{k+1}$

Gradient of the Objective Function: $\nabla \mathcal{L}_{\theta_c} = \sum_{t=1}^{T} \nabla_{\theta_c} \log \pi(a_t|s_t, \theta_c) R_t$

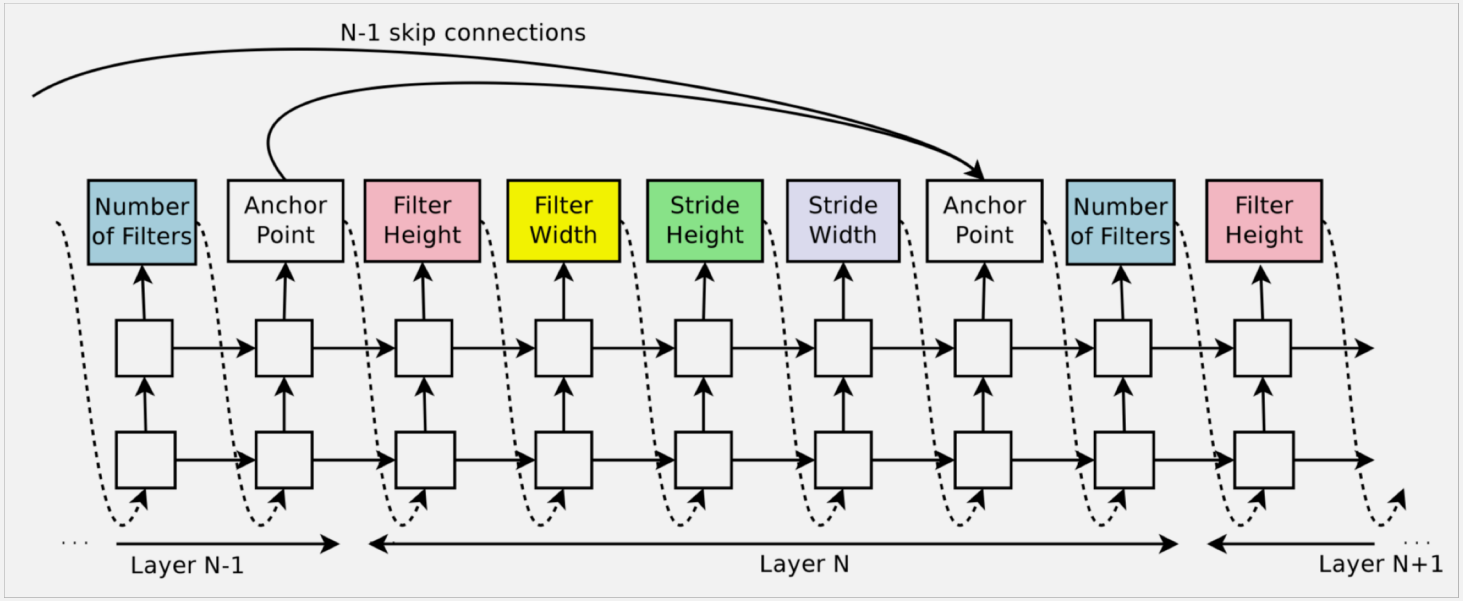
For one child network: $\nabla \mathcal{L}_{\theta_c} = \sum_{t=1}^{T} \nabla_{\theta_c} \log \pi(a_t|s_t, \theta_c) R$

For m child networks with baseline: $\nabla \mathcal{L}_{\theta_c} = \frac{1}{m} \sum_{k=1}^{m} \sum_{t=1}^{T} \nabla_{\theta_c} \log \pi(a_t|s_t, \theta_c)(R_k - b)$

CNN: Skip Connections

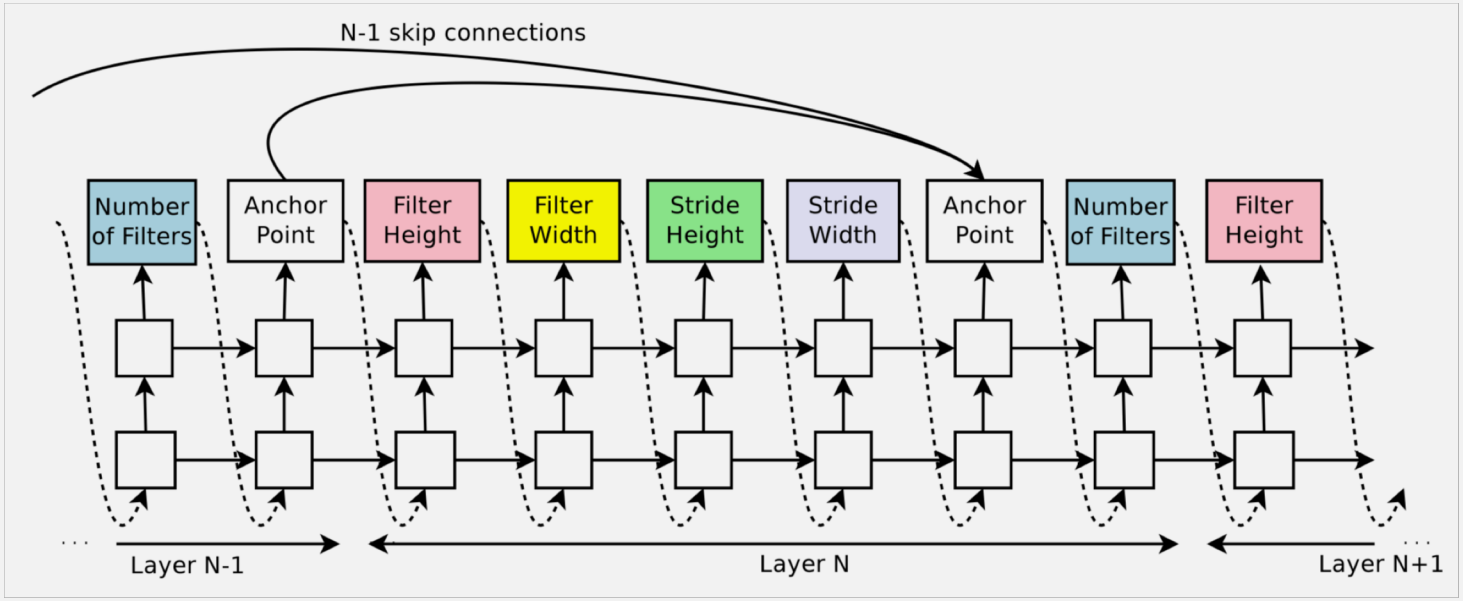
- Skip connections widen the search space
- Probability of anchor point j and i to be connected:

$$\mathrm{P}(\text{Layer j in input to layer i}) = \mathrm{sigmoid}(v^T \tanh(W_{prev} * h_j + W_{curr} * h_i))$$



CNN: Skip Connections – Compilation Failures

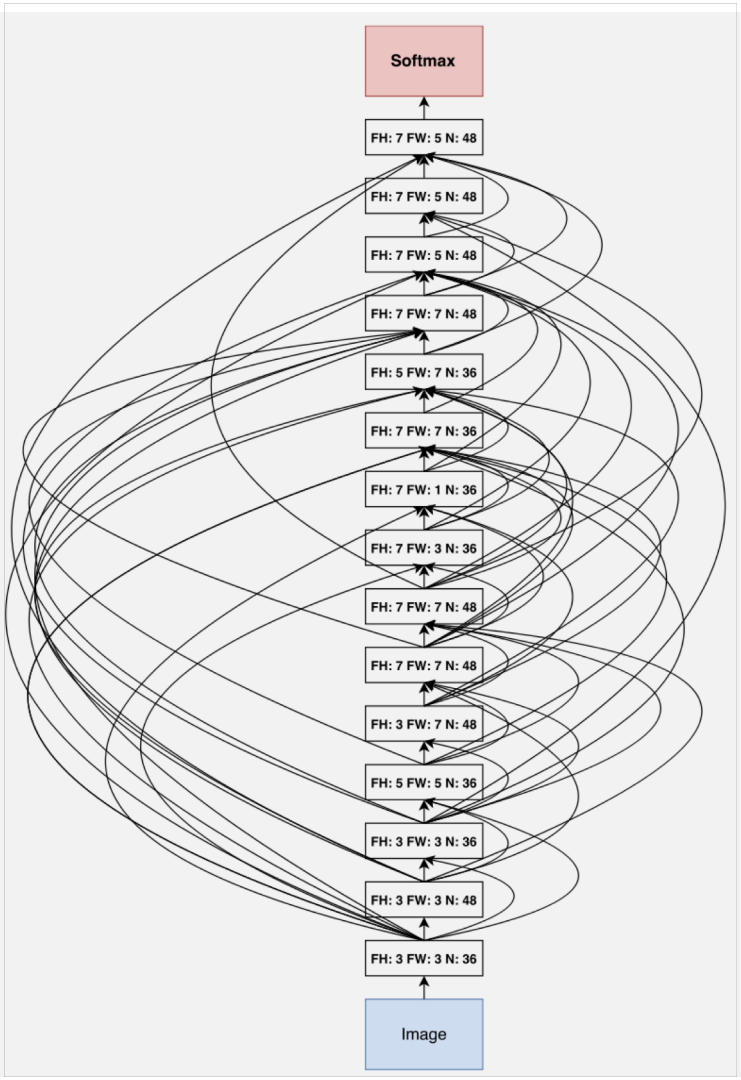
- Anchor has no input connection → Use image as input
- Anchor has no output → Connect anchor to the final layer
- Different input sizes → Pad all inputs to the same size



Experimental Results for CNN on CIFAR-10

- Controller is a two layer LSTM
- 800 networks are trained on 800 CPUs concurrently at any time
- After training 12 800 architectures, they selected the one with the best validation score
- Best architecture is then optimized by a hyperparameter grid search
- Transfer learning is not discussed

Experimental Results for CNN on CIFAR-10



Experimental Results for CNN on CIFAR-10

Model	Depth	Parameters	Error rate (%)
Network in Network (Lin et al., 2013)	-	-	8.81
All-CNN (Springenberg et al., 2014)	-	-	7.25
Deeply Supervised Net (Lee et al., 2015)	-	-	7.97
Highway Network (Srivastava et al., 2015)	-	-	7.72
Scalable Bayesian Optimization (Snoek et al., 2015)	-	-	6.37
FractalNet (Larsson et al., 2016)	21	38.6M	5.22
with Dropout/Drop-path	21	38.6M	4.60
ResNet (He et al., 2016a)	110	1.7M	6.61
ResNet (reported by Huang et al. (2016c))	110	1.7M	6.41
ResNet with Stochastic Depth (Huang et al., 2016c)	110	1.7M	5.23
	1202	10.2M	4.91
Wide ResNet (Zagoruyko & Komodakis, 2016)	16	11.0M	4.81
	28	36.5M	4.17
ResNet (pre-activation) (He et al., 2016b)	164	1.7M	5.46
	1001	10.2M	4.62
DenseNet $(L = 40, k = 12)$ Huang et al. (2016a)	40	1.0M	5.24
DenseNet$(L = 100, k = 12)$ Huang et al. (2016a)	100	7.0M	4.10
DenseNet $(L = 100, k = 24)$ Huang et al. (2016a)	100	27.2M	3.74
DenseNet-BC $(L = 100, k = 40)$ Huang et al. (2016b)	190	25.6M	3.46
Neural Architecture Search v1 no stride or pooling	15	4.2M	5.50
Neural Architecture Search v2 predicting strides	20	2.5M	6.01
Neural Architecture Search v3 max pooling	39	7.1M	4.47
Neural Architecture Search v3 max pooling + more filters	39	37.4M	3.65

Finding a RNN Cell with a RNN Controller

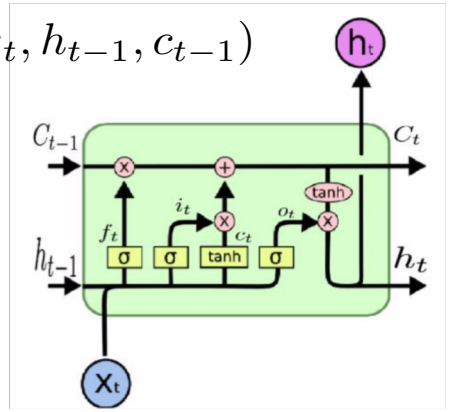
Sampling RNN 1: Model a RNN Cell Architecture

Simple RNN Cell

$$h_t = \tanh(W_1 * x_t + W_2 * h_{t-1})$$

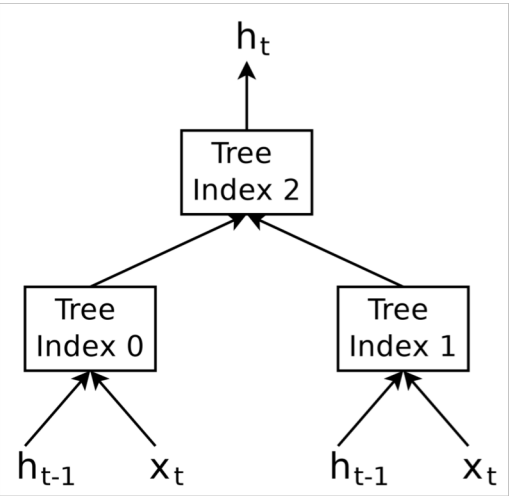
LSTM Cell

$$h_t, c_t = f(x_t, h_{t-1}, c_{t-1})$$

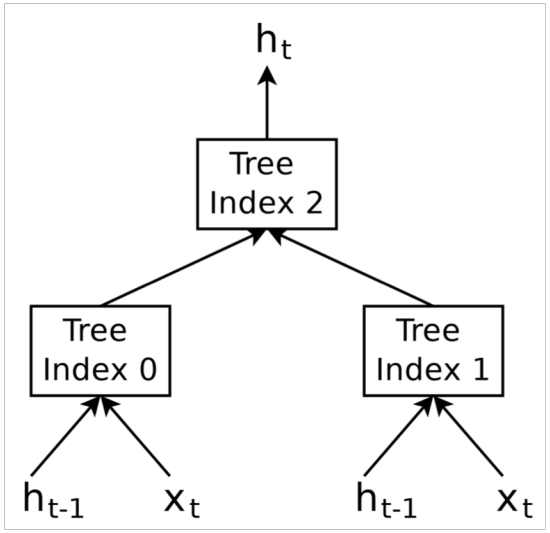


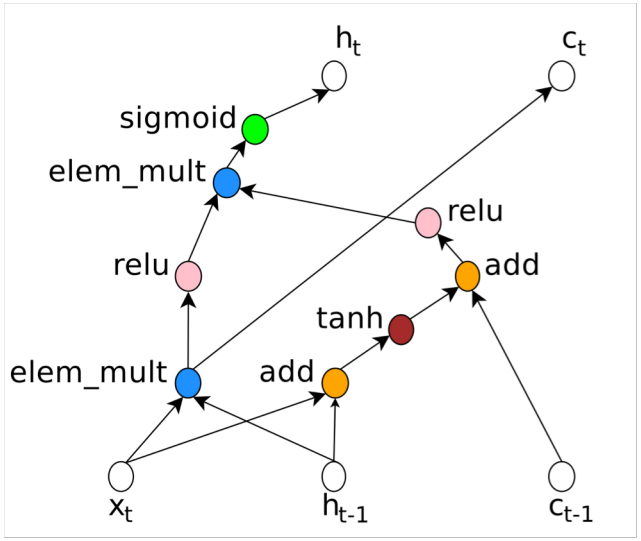
→ Model Cells by a tree structure with:
- Combination Method
- Activation Function

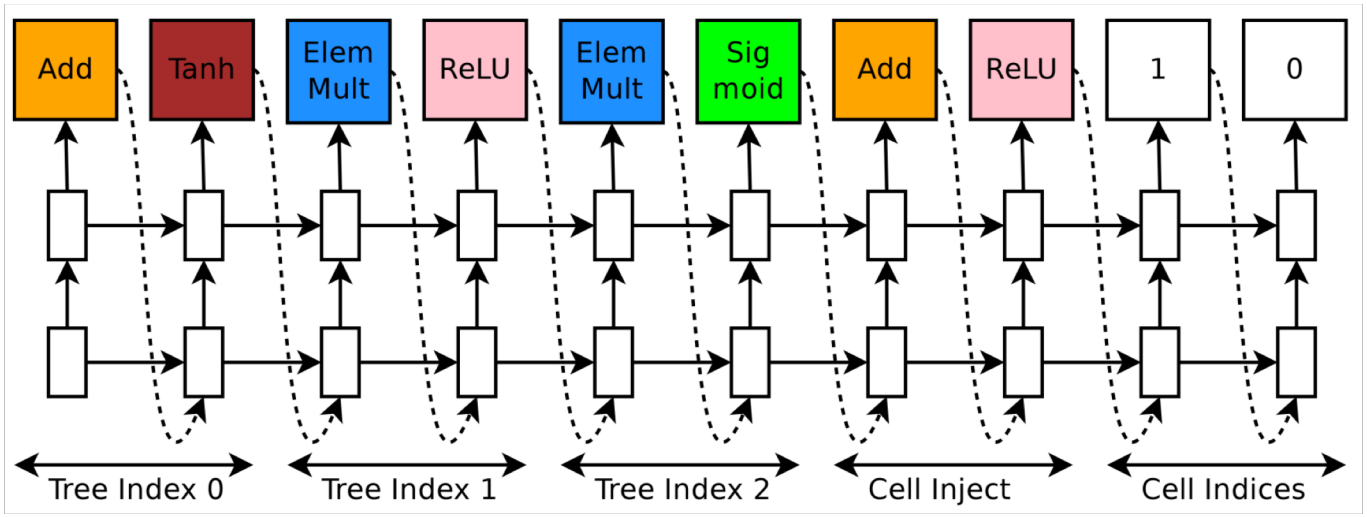
→ Manually selected base number determines the cell complexity



Sampling RNN 2: Sampling of the tree

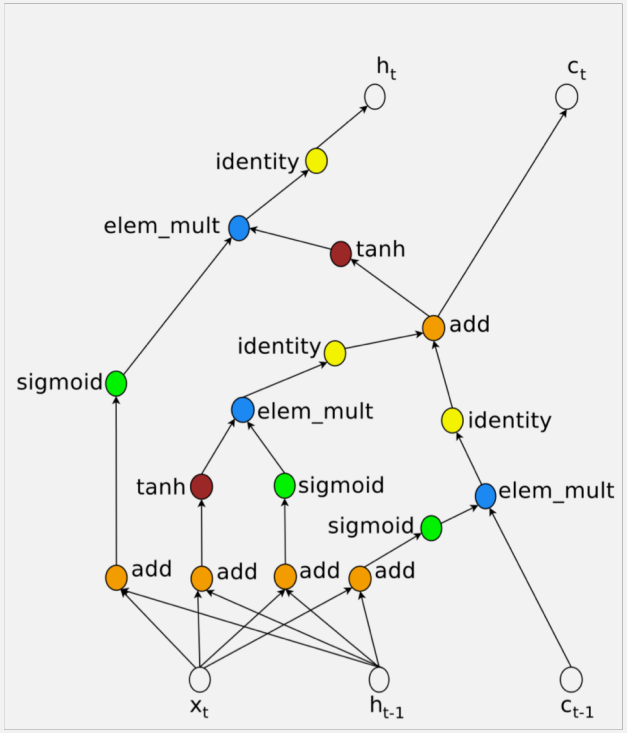




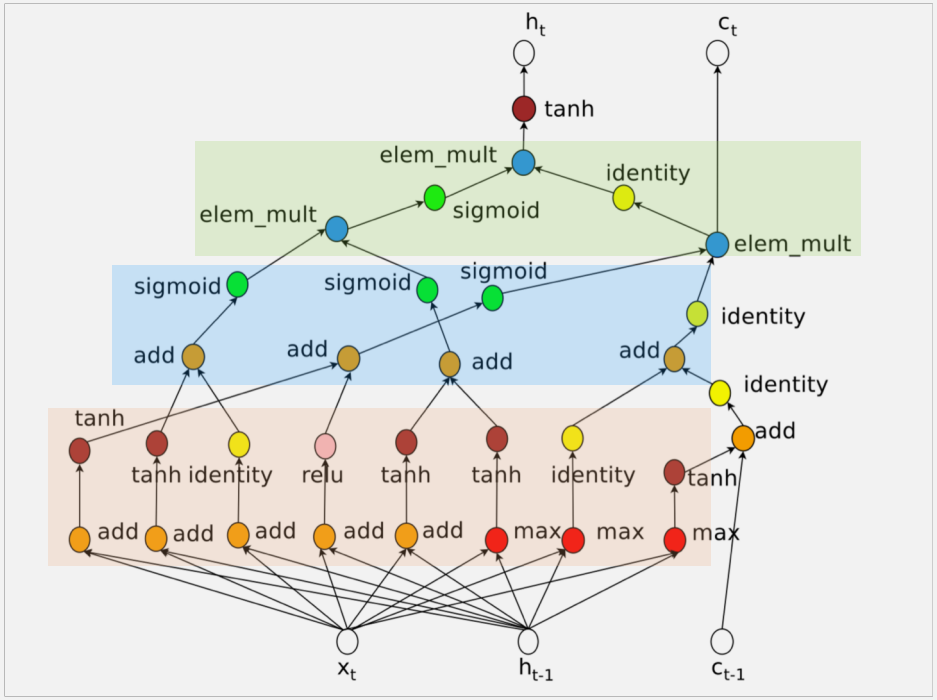


RNN - Finding new Cell Architectures

LSTM Cell



NAS Cell



Experimental Results for RNN on Penn Treebank

- Penn Treebank is a benchmark for language modelling
- 400 networks are trained on 400 CPUs concurrently at any time, in total 15,000 networks were trained
- Grid search on the best cell
- Improved state of the art perplexity
- Transfer Learning:
  - Use of the cell on the same dataset for a different task
  - Transfer to GNMT framework, on the WMT'14 English-German translation dataset

Experimental Results for RNN on Penn Treebank

Model	Parameters	Test Perplexity
Mikolov & Zweig (2012) - KN-5	2M‡	141.2
Mikolov & Zweig (2012) - KN5 + cache	2M‡	125.7
Mikolov & Zweig (2012) - RNN	6M‡	124.7
Mikolov & Zweig (2012) - RNN-LDA	7M‡	113.7
Mikolov & Zweig (2012) - RNN-LDA + KN-5 + cache	9M‡	92.0
Pascanu et al. (2013) - Deep RNN	6M	107.5
Cheng et al. (2014) - Sum-Prod Net	5M‡	100.0
Zaremba et al. (2014) - LSTM (medium)	20M	82.7
Zaremba et al. (2014) - LSTM (large)	66M	78.4
Gal (2015) - Variational LSTM (medium, untied)	20M	79.7
Gal (2015) - Variational LSTM (medium, untied, MC)	20M	78.6
Gal (2015) - Variational LSTM (large, untied)	66M	75.2
Gal (2015) - Variational LSTM (large, untied, MC)	66M	73.4
Kim et al. (2015) - CharCNN	19M	78.9
Press & Wolf (2016) - Variational LSTM, shared embeddings	51M	73.2
Merity et al. (2016) - Zoneout + Variational LSTM (medium)	20M	80.6
Merity et al. (2016) - Pointer Sentinel-LSTM (medium)	21M	70.9
Inan et al. (2016) - VD-LSTM + REAL (large)	51M	68.5
Zilly et al. (2016) - Variational RHN, shared embeddings	24M	66.0
Neural Architecture Search with base 8	32M	67.9
Neural Architecture Search with base 8 and shared embeddings	25M	64.0
Neural Architecture Search with base 8 and shared embeddings	54M	62.4

Strengths and Weaknesses

+ Neural Architecture Search introduced a new research direction
  - AutoAugment: Learning Augmentation Policies from Data, Cubuk et.al.
  - Searching for Activation functions, Ramachandran et.al.
  - Learning Transferable Architectures for Scalable Image Recognition, Zoph et.al.
+ Generalization of the RNN Cell

- Fix of 50 training epochs
- Wide search space and extreme resources
- Transfer learning for CNNs is not addressed