



# ST-NAS

## Neural Architecture Search for End-to-End Speech Recognition via Straight-Through Gradients 基于直通梯度的端到端语音识别神经架构搜索

欧智坚

Speech Processing and Machine Intelligence (SPMI) Lab, Tsinghua University

<http://oa.ee.tsinghua.edu.cn/ouzhijian/>

SLT 2021 Children Speech Recognition Challenge (CSRC) Workshop, 2021/1/30

# Outline

- CTC-CRF: 基于CTC拓扑的条件随机场

- ✓ ICASSP 2019, INTERSPEECH 2020
- ✓ 开源CAT (Crf based Asr Toolkit) <https://github.com/thu-spmi/CAT>
- ✓ 综合了混合系统和端到端系统的优势，在保持端到端系统简洁性同时，实现了数据高效以及低延迟流式语音识别，在AISHELL中文语音识别错误率达到了据我们所知最低的6.34% (4-gram LM) !

- ST-NAS: 基于直通梯度的神经网络架构搜索

- ✓ SLT 2021
- ✓ 开源 <https://github.com/thu-spmi/ST-NAS>
- ✓ 面对应接不暇的神经网络架构，不妨试试我们提出的ST-NAS，在WSJ eval92英文语音识别错误率达到了据我们所知最低的2.77% (4-gram LM) !

# CTC-CRF

- 向鸿雨, 欧智坚.  
CRF-based Single-stage Acoustic Modeling with CTC Topology. ICASSP 2019. [Oral]
- 安柯宇, 向鸿雨, 欧智坚.  
CAT: A CTC-CRF based ASR Toolkit Bridging the Hybrid and the End-to-end Approaches towards Data Efficiency and Low Latency. INTERSPEECH 2020.

# Content

1. Motivation

2. Related work

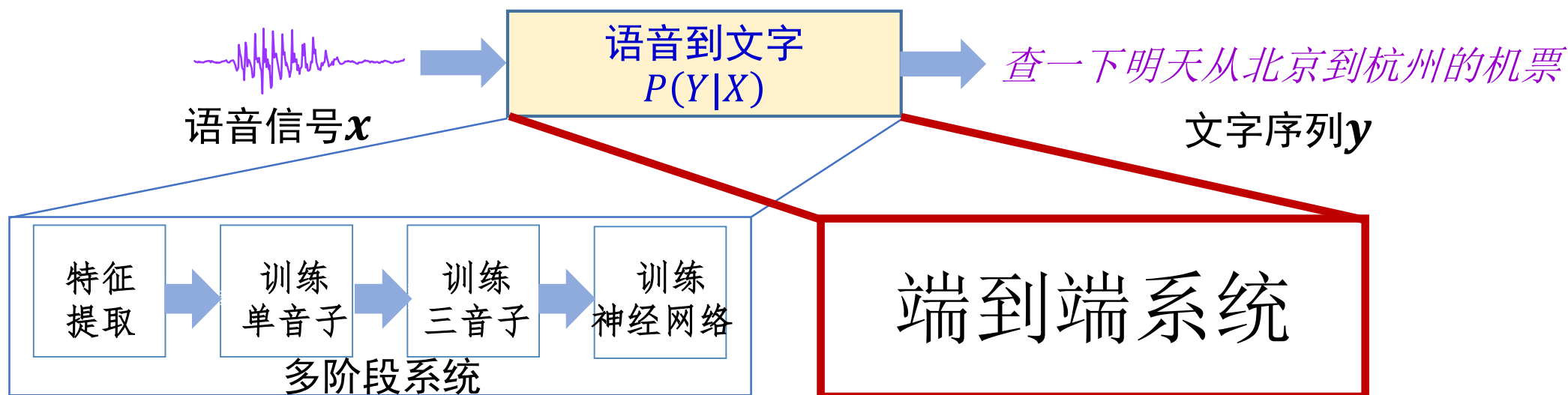
3. CTC-CRF

4. Experiments

5. Conclusion

# Introduction

- ASR is a discriminative problem
  - For acoustic observations  $\mathbf{x} \triangleq x_1, \dots, x_T$ , find the most likely labels  $\mathbf{y} \triangleq y_1, \dots, y_L$
- ASR state-of-the-art: DNNs of various network architectures (Hinton NIPSw2009, Microsoft IS2011)



# Motivation

- End-to-end system:

- Eliminate GMM-HMM pre-training and tree building, and can be trained from scratch (flat-start or single-stage).

- In a more strict sense:

- Remove the need for a pronunciation lexicon and, even further, train the acoustic and language models jointly rather than separately
- Data-hungry

We advocate data-efficient end2end speech recognition, which uses a separate language model (LM) with or without a pronunciation lexicon.

- Text corpus for language modeling are cheaply available.
- Data-efficient

# Related work

## ASR is a discriminative problem

- For acoustic observations  $\mathbf{x} \triangleq x_1, \dots, x_T$ , find the most likely labels  $\mathbf{y} \triangleq y_1, \dots, y_L$

### 1. How to obtain $p(\mathbf{y} | \mathbf{x})$

### 2. How to handle alignment, since $L \neq T$

- Explicitly by state sequence  $\boldsymbol{\pi} \triangleq \pi_1, \dots, \pi_T$  in HMM, CTC, RNN-T, or implicitly in Seq2Seq

Labels

$L \neq T$

$\mathbf{y}$								
$\parallel$						$\pi_7$	$\pi_8$	
$y_1$						$\pi_6$		
$\vdots$								
$y_L$	$\pi_1$	$\pi_2$						

Observations  $\mathbf{x} = x_1 \dots x_T$

# Related work How to handle alignment, since $L \neq T$

- Explicitly by state sequence  $\boldsymbol{\pi} \triangleq \pi_1, \dots, \pi_T$  in HMM, CTC, RNN-T, or implicitly in Seq2Seq
- State topology : determines a mapping  $\mathcal{B}$ , which map  $\boldsymbol{\pi}$  to a unique  $l$

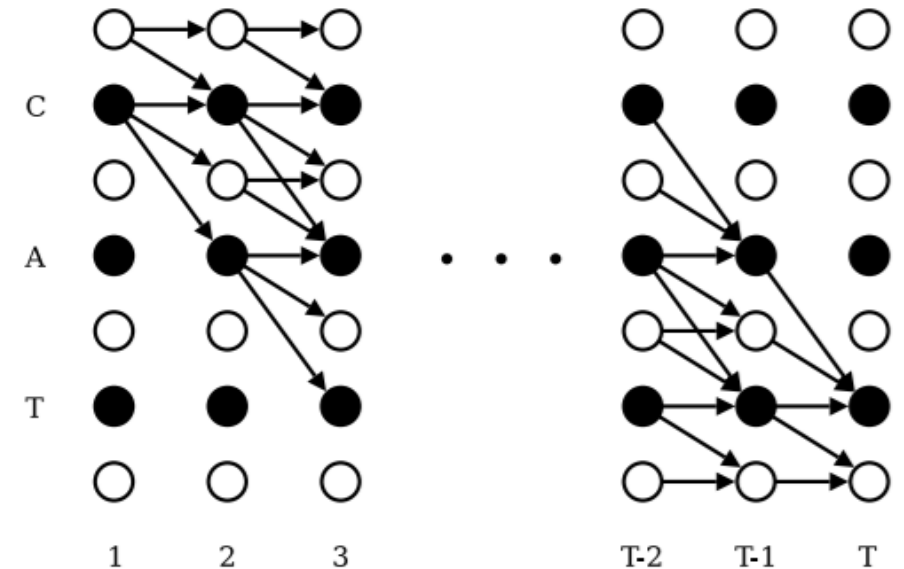
$$p(\mathbf{y}|\mathbf{x}) = \sum_{\boldsymbol{\pi} \in \mathcal{B}^{-1}(l)} p(\boldsymbol{\pi}|\mathbf{x})$$

CTC topology : a mapping  $\mathcal{B}$  maps  $\boldsymbol{\pi}$  to  $l$  by

1. removing all repetitive symbols between the blank symbols.
2. removing all blank symbols.

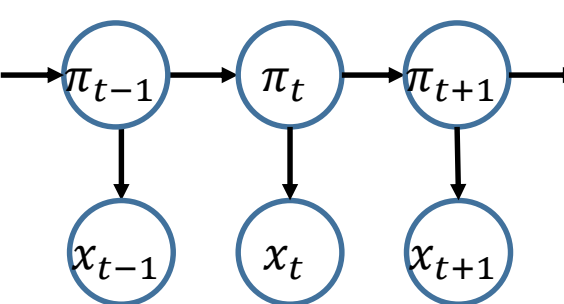
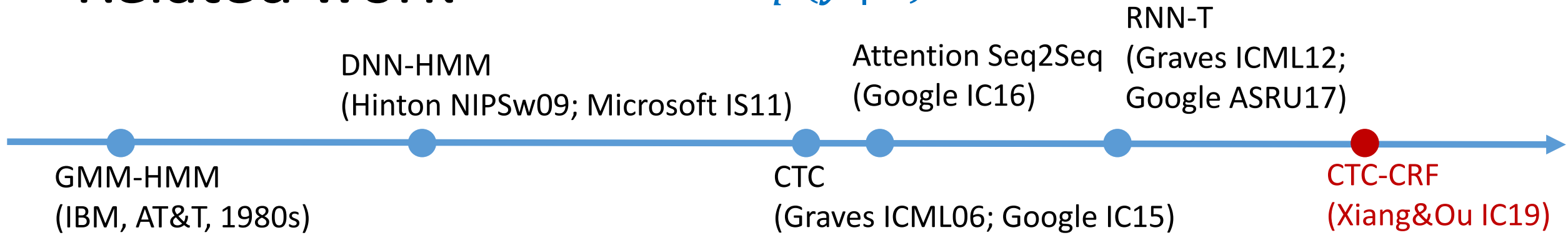
$$\mathcal{B}(-CC - -AA - T -) = CAT$$

- ☺ Admit the smallest number of units in state inventory, by adding only one `<blk>` to label inventory.
- ☺ Avoid ad-hoc silence insertions in estimating denominator LM of labels.

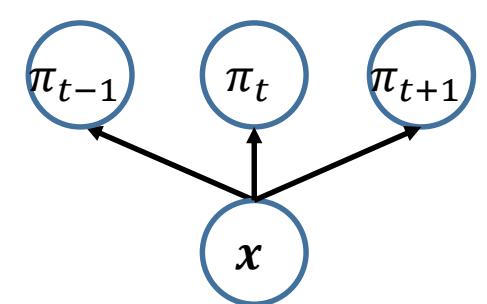




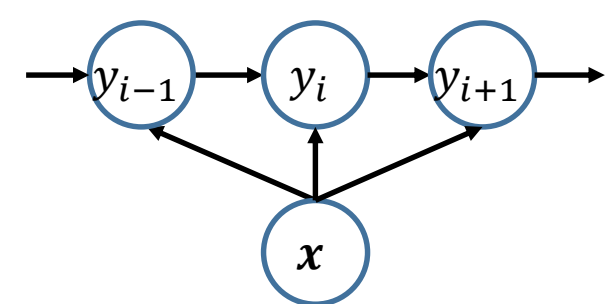
# Related work How to obtain $p(\mathbf{y} | \mathbf{x})$



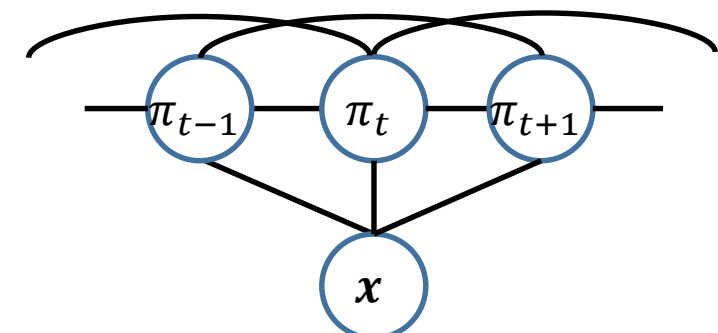
**DNN-HMM**  
缺陷: 多阶段



**CTC**  
缺陷:  $\{\pi_t\}$ 条件独立性



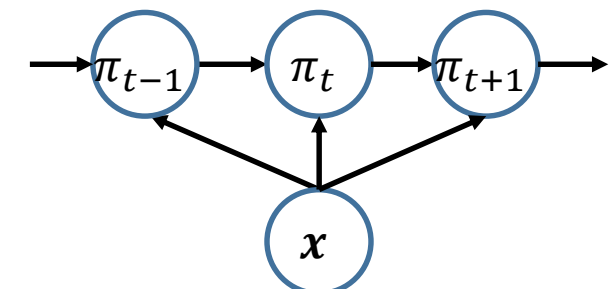
**Attention**  
缺陷:  $\{y_i\}$ 有向图序列模型



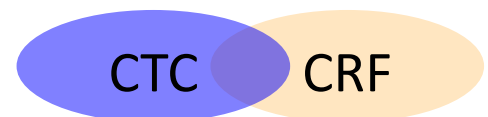
**CTC-CRF**

历史上各类模型具有不同的图结构，  
CTC-CRF占有独特位置！

$$p(\mathbf{y}|\mathbf{x}) = \sum_{\boldsymbol{\pi} \in \mathcal{B}^{-1}(\mathbf{l})} p(\boldsymbol{\pi}|\mathbf{x})$$

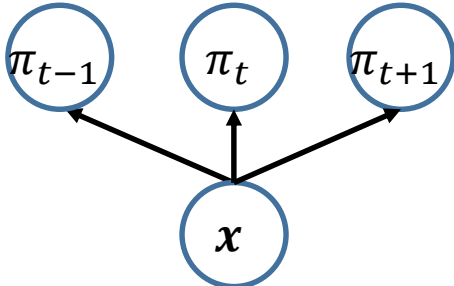
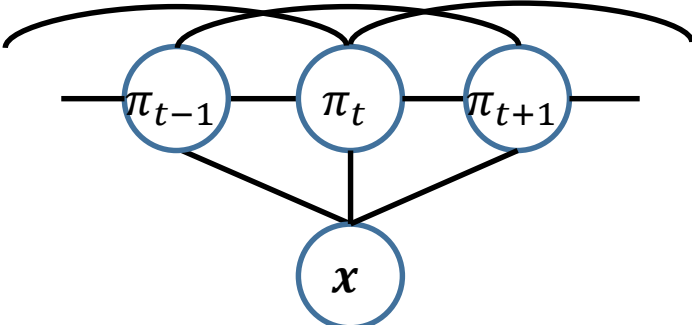


**RNN-T**  
缺陷:  $\{\pi_t\}$ 有向图序列模型



联合神经网络与无向图模型

# CTC vs CTC-CRF

CTC	CTC-CRF
$p(\mathbf{l} \mathbf{x}) = \sum_{\boldsymbol{\pi} \in \mathcal{B}^{-1}(\mathbf{l})} p(\boldsymbol{\pi} \mathbf{x}), \text{ using CTC topology } \mathcal{B}$	
<p>State Independence</p> $p(\boldsymbol{\pi} \mathbf{x}; \boldsymbol{\theta}) = \prod_{t=1}^T p(\pi_t \mathbf{x})$	$p(\boldsymbol{\pi} \mathbf{x}; \boldsymbol{\theta}) = \frac{e^{\phi(\boldsymbol{\pi}, \mathbf{x}; \boldsymbol{\theta})}}{\sum_{\boldsymbol{\pi}'} e^{\phi(\boldsymbol{\pi}', \mathbf{x}; \boldsymbol{\theta})}}$ <p style="text-align: right; color: red;">Node potential, by NN</p> $\phi(\boldsymbol{\pi}, \mathbf{x}; \boldsymbol{\theta}) = \sum_{t=1}^T \left( \log p(\pi_t \mathbf{x}) + \log p_{LM}(\mathcal{B}(\boldsymbol{\pi})) \right)$ <p style="text-align: right; color: red;">Edge potential, by n-gram denominator LM of labels, like in LF-MMI</p>
$\frac{\partial \log p(\mathbf{l} \mathbf{x}; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = \mathbb{E}_{p(\boldsymbol{\pi} \mathbf{l}, \mathbf{x}; \boldsymbol{\theta})} \left[ \frac{\partial \log p(\boldsymbol{\pi} \mathbf{x}; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \right]$	$\frac{\partial \log p(\mathbf{l} \mathbf{x}; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = \mathbb{E}_{p(\boldsymbol{\pi} \mathbf{l}, \mathbf{x}; \boldsymbol{\theta})} \left[ \frac{\partial \phi(\boldsymbol{\pi}, \mathbf{x}; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \right] - \mathbb{E}_{p(\boldsymbol{\pi}' \mathbf{x}; \boldsymbol{\theta})} \left[ \frac{\partial \phi(\boldsymbol{\pi}', \mathbf{x}; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \right]$
	

# SS-LF-MMI vs CTC-CRF

	SS-LF-MMI	CTC-CRF
State topology	HMM topology with two states	CTC topology
Silence label	Using silence labels. Silence labels are randomly inserted when estimating denominator LM.	No silence labels. Use <blk> to absorb silence. 😊 No need to insert silence labels to transcripts.
Decoding	No spikes.	The posterior is dominated by <blk> and non-blank symbols occur in spikes. 😊 Speedup decoding by skipping blanks.
Implementation	Modify the utterance length to one of 30 lengths; use leaky HMM.	😊 No length modification; no leaky HMM.

# Experiments

- We conduct our experiments on three benchmark datasets:
  - WSJ 80 hours
  - Switchboard 300 hours
  - Librispeech 1000 hours
- **Acoustic model:** 6 layer BLSTM with **320** hidden dim, 13M parameters
- **Adam optimizer** with an initial learning rate of 0.001, decreased to 0.0001 when cv loss does not decrease
- **Implemented with Pytorch.**
- **Objective function** (use the CTC objective function to help convergences):

$$\mathcal{J}_{CTC-CRF} + \alpha \mathcal{J}_{CTC}$$

- **Decoding score function** (use word-based language models, WFST based decoding):

$$\log p(\mathbf{l}|\mathbf{x}) + \beta \log p_{LM}(\mathbf{l})$$

# Experiments (Comparison with CTC, phone based)

## WSJ 80h

Model	Unit	LM	SP	dev93	eval92
CTC	Mono-phone	4-gram	N	10.81%	7.02%
CTC-CRF	Mono-phone	4-gram	N	6.24%	3.90%

44.4% reduction in eval92 error rate for CTC-CRF compared to CTC.

## Switchboard 300h

Model	Unit	LM	SP	SW	CH
CTC	Mono-phone	4-gram	N	12.9%	23.6%
CTC-CRF	Mono-phone	4-gram	N	11.0%	21.0%

14.7% reduction in SW error rate and 11% reduction in CH error rate for CTC-CRF compared to CTC.

## Librispeech 1000h

Model	Unit	LM	SP	Dev Clean	Dev Other	Test Clean	Test Other
CTC	Mono-phone	4-gram	N	4.64%	13.23%	5.06%	13.68%
CTC-CRF	Mono-phone	4-gram	N	3.87%	10.28%	4.09%	10.65%

19.1% reduction in Test Clean error rate and 22.1% reduction in Test Other error rate for CTC-CRF compared to CTC.

SP: speed perturbation for 3-fold data augmentation.

# Experiments (Comparison with STOA)

## Switchboard 300h

Model	SW	CH	Average	Source
Kaldi chain triphone	9.6	19.3	14.5	IS 2016
Kaldi e2e chain monophone	11.0	20.7	15.9	ASLP 2018, 26M
Kaldi e2e chain biphone	9.8	19.3	14.6	ASLP 2018, 26M
CTC-CRF monophone	10.3	19.7	15.0	ICASSP 2019, BLSTM, 13M
<b>CTC-CRF monophone</b>	<b>9.8</b>	<b>18.8</b>	<b>14.3</b>	<b>IS 2020, VGG BLSTM, 16M</b>
DNN-HMM triphone	9.8	19.0	14.4	RWTH IS 2018
DNN-HMM triphone	9.6	19.3	14.5	IBM IS 2019
Seq2Seq subword	11.8	25.7	18.8	RWTH IS 2018, LSTM-LM
Seq2Seq subword	10.7	20.7	15.7	Espnet ASRU19

10%

RWTH IS 2018, “Improved training of end-to-end attention models for speech recognition”.

RWTH IS 2019, “RWTH ASR Systems for LibriSpeech Hybrid vs Attention -- Data Augmentation”.

IBM IS19, “Forget a Bit to Learn Better Soft Forgetting for CTC-based Automatic Speech Recognition”.

Espnet ASRU19, “Espresso: A Fast End-to-end Neural Speech Recognition Toolkit”.

Google IS19, “SpecAugment: A Simple Data Augmentation Method for Automatic Speech Recognition”.

# Experiments (Comparison with STOA)

## Librispeech 1000h

Model	Test Clean	Test Other	Source
Kaldi chain triphone	4.28	-	IS 2016
<b>CTC-CRF monophone</b>	<b>4.0</b>	<b>10.6</b>	<b>ICASSP 2019, BLSTM(6,320), 13M</b>
DNN-HMM triphone	4.4	10.0	RWTH IS 2019
Seq2Seq subword	4.8	15.3	RWTH IS 2018
Seq2Seq subword	4.0	12.0	Espnet ASRU19
Seq2Seq subword	4.1	12.5	Google IS19 (w/o SpecAugment)

RWTH IS 2018, “Improved training of end-to-end attention models for speech recognition”.

RWTH IS 2019, “RWTH ASR Systems for LibriSpeech Hybrid vs Attention -- Data Augmentation”.

IBM IS19, “Forget a Bit to Learn Better Soft Forgetting for CTC-based Automatic Speech Recognition”.

Espnet ASRU19, “Espresso: A Fast End-to-end Neural Speech Recognition Toolkit”.

Google IS19, “SpecAugment: A Simple Data Augmentation Method for Automatic Speech Recognition”.

# Aishell-1 results

- 170 hours mandarin speech corpus
- 400 speakers from different accent areas
- 15% CER reduction compared with LF-MMI
- 5% CER reduction compared with end-to-end transformer

Model	%CER
LF-MMI with i-vector [1]	7.43
Transformer [2]	6.7
CTC-CRF [3]	6.34

[1] D. Povey, A. Ghoshal, and et al, "The Kaldi speech recognition toolkit," ASRU 2011.

[2] S. Karita, N. Chen, and et al, "A comparative study on transformer vs RNN in speech applications," ASRU 2019.

[3] Keyu An, Hongyu Xiang, and Zhijian Ou, "CAT: A CTC-CRF based ASR toolkit bridging the hybrid and the end-to-end approaches towards data efficiency and low latency," INTERSPEECH 2020.



# 2021 SLT CHILDREN SPEECH RECOGNITION CHALLENGE (CSRC)

ORGANIZER :  西北工业大学  清华大学  厦門大學  标贝科技 

- 400 hours of data, targeting to boost children speech recognition research.
- Evaluated on 10 hours of children's reading and conversational speech.
- 3 baselines (Chain model, Transformer and CTC-CRF) are provided.

model	Chain model	Transformer	CTC-CRF
CER%	28.75	27.28	<b>25.34</b>

Fan Yu, Zhuoyuan Yao, Xiong Wang, Keyu An, Lei Xie, Zhijian Ou, Bo Liu, Xiulin Li, Guanqiong Miao. The SLT 2021 children speech recognition challenge: Open datasets, rules and baselines. SLT 2021.

# 基于条件随机场的高效端到端语音识别 – 总结

- 在WSJ、Switchboard、Librispeech, 性能表现均超过了
  - CTC、Attention Seq2Seq (15%相对改进),
  - 现在广为流行的Kaldi工具包中的端对端模型e2e Chain-model (10%相对改进),
  - 与多阶段Chain-model持平,
- 同时具有训练流程简洁、能充分利用词典及语言模型从而数据利用效率高等优势。

# 开源 Crf-based Asr Toolkit (CAT)



## 1. CAT contains a full-fledged implementation of CTC-CRF

- Fast parallel calculation of gradients using CUDA C/C++ interface

## 2. CAT adopts PyTorch to build DNNs

## 3. CAT provides a complete workflow with example recipes

## 4. Flexibility and future work

- ✓ Streaming ASR <- INTRESPEECH 2020
- CRFs with different topologies (e.g., RNN-T)
- Multi-lingual, Code-mix
- ✓ NAS <- SLT 2021
- ...



# ST-NAS

- 郑华焕, 安柯宇, 欧智坚. Efficient Neural Architecture Search for End-to-end Speech Recognition via Straight-Through Gradients. SLT 2021.

# Content

1. Motivation

2. Related work

3. ST-NAS

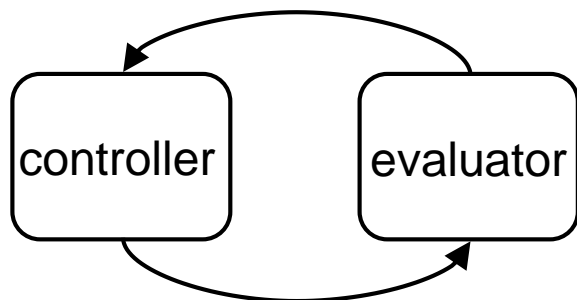
4. Experiments

5. Conclusion

# Motivation

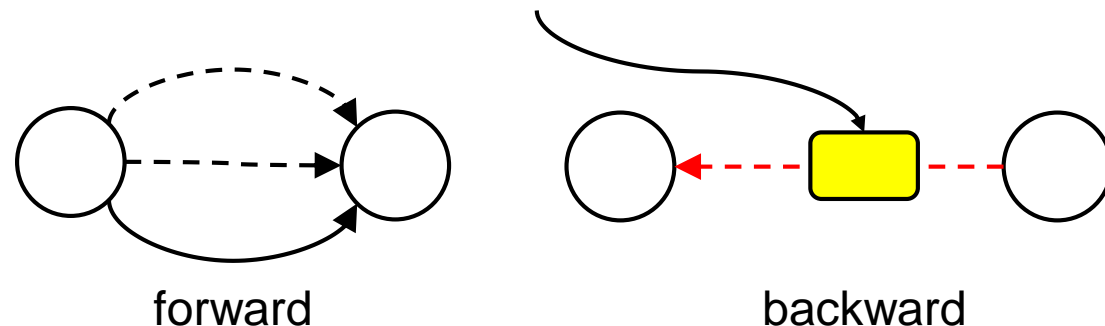
- End-to-end ASR reduces expert efforts by automating *feature engineering*, but raises a demand for *architecture engineering*.
- Neural architecture search (NAS), the process of automating architecture engineering, is an appealing next step to advancing end-to-end ASR.

## 1. Early NAS methods



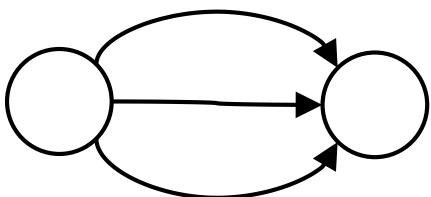
computation expensive,  
1000+ GPU days 😞

## 3. (ours) Straight-Through gradient NAS (ST-NAS)



Back-Prop ST gradients through the sampled edge,  
**Efficient** in both memory and computation,  
Less than 3-fold computation time 😊

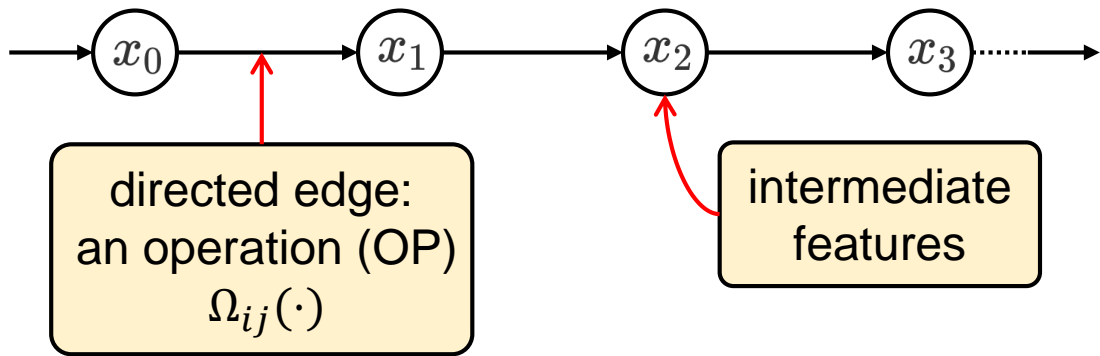
## 2. Recent gradient-based NAS methods (DARTS, SNAS, ProxylessNAS)



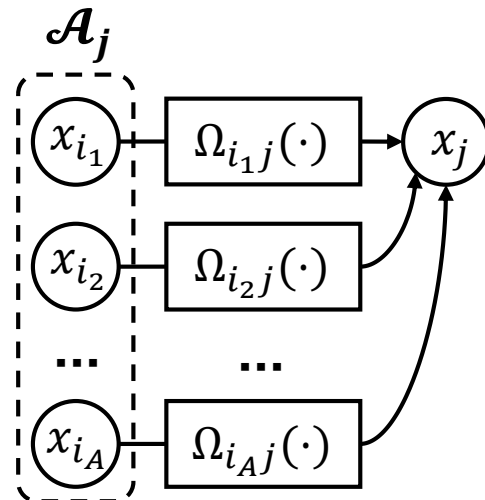
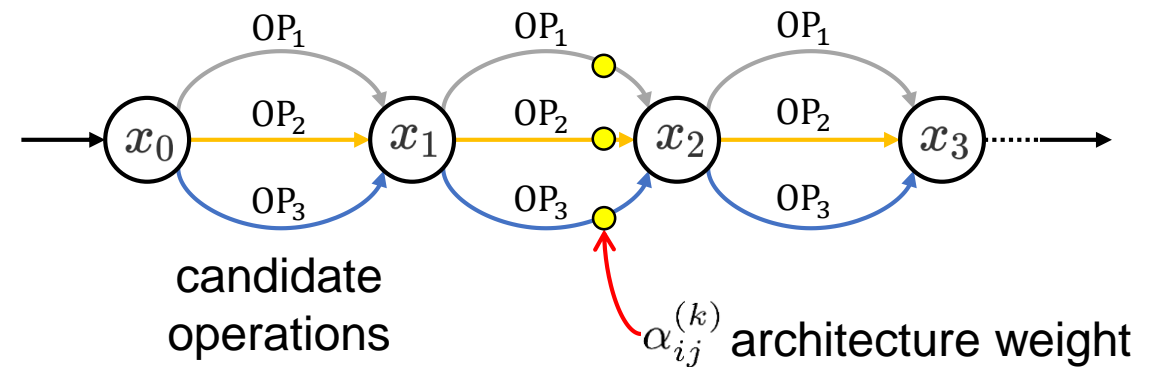
Improved,  
but still memory expensive (DARTS, SNAS),  
or using ad-hoc trick (ProxylessNAS) 😞

# Gradient-based NAS: representing the search space as a weighted directed acyclic graph (DAG)

NN architecture as graph



The DAG of NAS: "supernet"

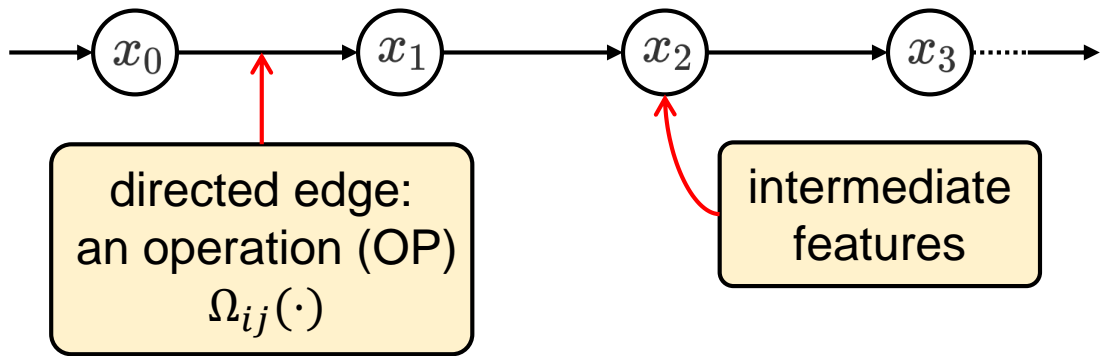


Forward computation in general

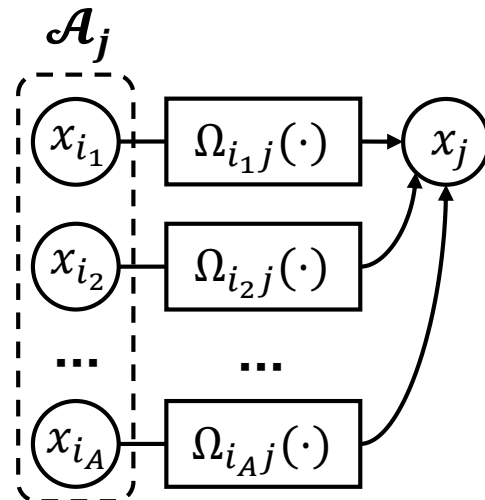
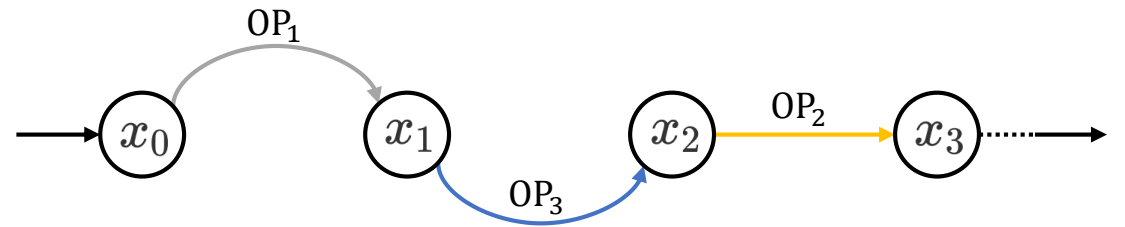
$$\Rightarrow x_j = \sum_{i \in \mathcal{A}_j} \Omega_{ij}(x_i)$$

# Gradient-based NAS: representing the search space as a weighted directed acyclic graph (DAG)

NN architecture as graph



The sampled sub-graph



Forward computation in general

$$\Rightarrow x_j = \sum_{i \in \mathcal{A}_j} \Omega_{ij}(x_i)$$

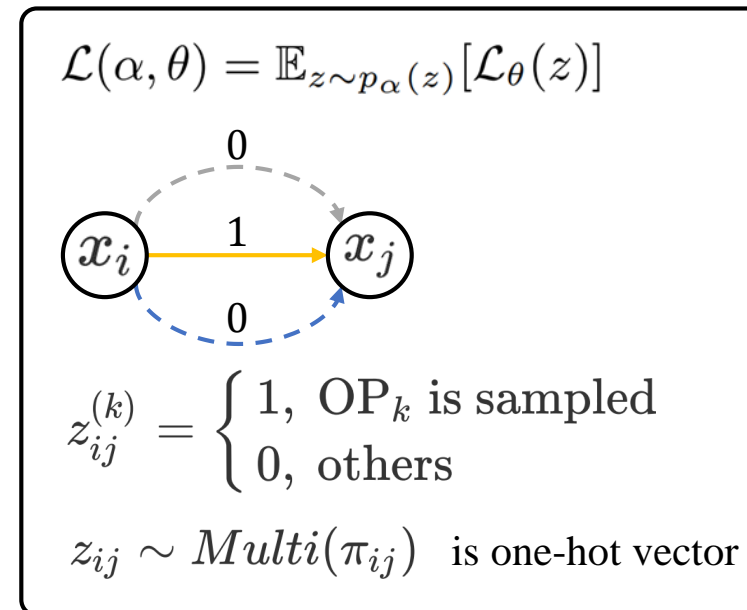


# Related work: DARTS, SNAS and ProxylessNAS

DARTS

Definition	$\Omega_{ij}^{\text{DARTS}}(x_i) = \sum_{k=1}^K \pi_{ij}^{(k)} o_{ij}^{(k)}(x_i)$ $\pi_{ij}^{(k)} = \frac{\exp(\alpha_{ij}^{(k)})}{\sum_{k'=1}^K \exp(\alpha_{ij}^{(k')})}$
Limitation	<div style="border: 2px solid red; padding: 5px; display: inline-block;">                     Continuous relaxation;                      Require <math>K \times</math> memory and time                 </div>

Ideally



- [1] H Liu, K Simonyan, and Y Yang, “**DARTS**: Differentiable architecture search,” in ICLR, 2019.
- [2] S Xie, H Zheng, et al, “**SNAS**: stochastic neural architecture search,” in ICLR, 2019.
- [3] H Cai, L Zhu, and S Han, “**ProxylessNAS**: Direct neural architecture search on target task and hardware,” in ICLR, 2019.
- [4] E Jang, S Gu, and B Poole, “Categorical reparameterization with Gumbel-Softmax,” in ICLR, 2017.
- [5] M Courbariaux, Y Bengio, and J David, “BinaryConnect: training deep neural networks with binary weights during propagations,” NIPS, 2015.

# Related work: DARTS, SNAS and ProxylessNAS

	DARTS	SNAS
Definition	$\Omega_{ij}^{\text{DARTS}}(x_i) = \sum_{k=1}^K \pi_{ij}^{(k)} o_{ij}^{(k)}(x_i)$ $\pi_{ij}^{(k)} = \frac{\exp(\alpha_{ij}^{(k)})}{\sum_{k'=1}^K \exp(\alpha_{ij}^{(k')})}$	$\tilde{\mathcal{L}}(\alpha, \theta) = \mathbb{E}_{y \sim p_{\alpha}(y)} [\mathcal{L}_{\theta}(y)]$ $\Omega_{ij}^{\text{SNAS}}(x_i) = \sum_{k=1}^K y_{ij}^{(k)} o_{ij}^{(k)}(x_i)$ $y_{ij}^{(k)} = \text{GumbelSoftmax}(\alpha_{ij}, k)$
Limitation	<div style="border: 2px solid red; padding: 5px; display: inline-block;">                     Continuous relaxation;                      Require <math>K \times</math> memory and time                 </div>	Same as DARTS

- [1] H Liu, K Simonyan, and Y Yang, “**DARTS**: Differentiable architecture search,” in ICLR, 2019.
- [2] S Xie, H Zheng, et al, “**SNAS**: stochastic neural architecture search,” in ICLR, 2019.
- [3] H Cai, L Zhu, and S Han, “**ProxylessNAS**: Direct neural architecture search on target task and hardware,” in ICLR, 2019.
- [4] E Jang, S Gu, and B Poole, “Categorical reparameterization with Gumbel-Softmax,” in ICLR, 2017.
- [5] M Courbariaux, Y Bengio, and J David, “BinaryConnect: training deep neural networks with binary weights during propagations,” NIPS, 2015.

# Related work: DARTS, SNAS and ProxylessNAS

	DARTS	SNAS	ProxylessNAS
Definition	$\Omega_{ij}^{\text{DARTS}}(x_i) = \sum_{k=1}^K \pi_{ij}^{(k)} o_{ij}^{(k)}(x_i)$ $\pi_{ij}^{(k)} = \frac{\exp(\alpha_{ij}^{(k)})}{\sum_{k'=1}^K \exp(\alpha_{ij}^{(k')})}$	$\tilde{\mathcal{L}}(\alpha, \theta) = \mathbb{E}_{y \sim p_{\alpha}(y)} [\mathcal{L}_{\theta}(y)]$ $\Omega_{ij}^{\text{SNAS}}(x_i) = \sum_{k=1}^K y_{ij}^{(k)} o_{ij}^{(k)}(x_i)$ $y_{ij}^{(k)} = \text{GumbelSoftmax}(\alpha_{ij}, k)$	$\Omega_{ij}(x_i) = \sum_{k=1}^K z_{ij}^{(k)} o_{ij}^{(k)}(x_i)$ $z_{ij} \sim \text{Multi}(\pi_{ij})$
Limitation	<div style="border: 2px solid red; padding: 5px;">                     Continuous relaxation;                      Require <math>K \times</math> memory and time                 </div>	Same as DARTS	The loss is not explicitly shown in the original paper

[1] H Liu, K Simonyan, and Y Yang, “**DARTS**: Differentiable architecture search,” in ICLR, 2019.

[2] S Xie, H Zheng, et al, “**SNAS**: stochastic neural architecture search,” in ICLR, 2019.

[3] H Cai, L Zhu, and S Han, “**ProxylessNAS**: Direct neural architecture search on target task and hardware,” in ICLR, 2019.

[4] E Jang, S Gu, and B Poole, “Categorical reparameterization with Gumbel-Softmax,” in ICLR, 2017.

[5] M Courbariaux, Y Bengio, and J David, “BinaryConnect: training deep neural networks with binary weights during propagations,” NIPS, 2015.

# ST (Straight-Through) NAS

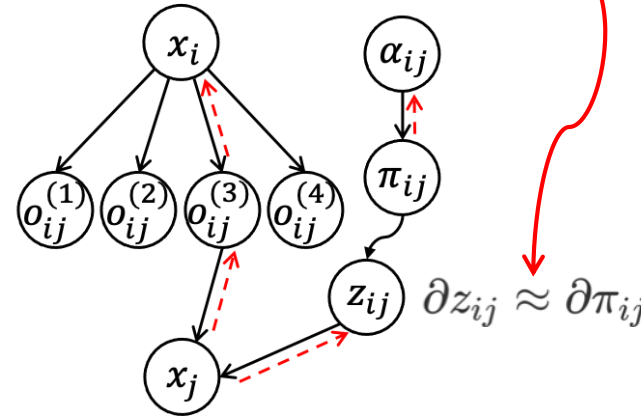
$$\mathcal{L}(\alpha, \theta) = \mathbb{E}_{z \sim p_{\alpha}(z)} [\mathcal{L}_{\theta}(z)]$$

$$\Omega_{ij}(x_i) = \sum_{k=1}^K z_{ij}^{(k)} o_{ij}^{(k)}(x_i)$$

$$z_{ij} \sim \text{Multi}(\pi_{ij})$$

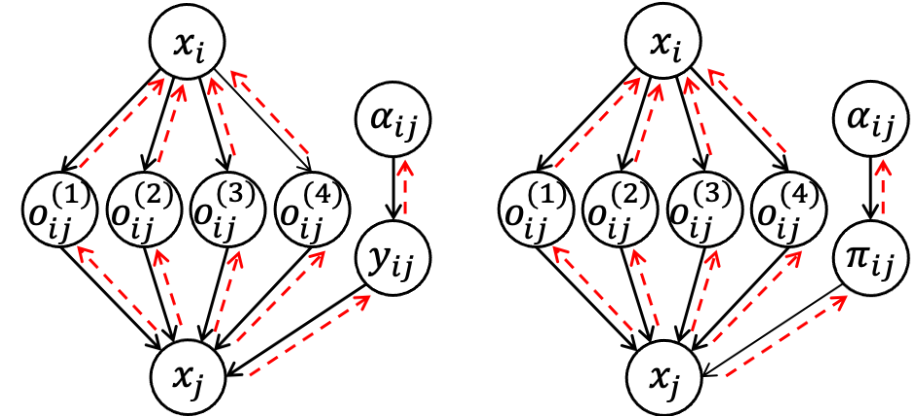
$$\pi_{ij}^{(k)} = \frac{\exp(\alpha_{ij}^{(k)})}{\sum_{k'=1}^K \exp(\alpha_{ij}^{(k')})}$$

straight-through gradient



continuous relaxation

SNAS ← DARTS



Using ST gradients to support sub-graph sampling is key to achieve efficient NAS beyond DARTS and SNAS.

Comparison of different gradient-based NAS methods.

Methods	Loss	$\alpha$ gradient	Memory	Backward computation
DARTS	$\mathcal{L}^{\text{DARTS}}(\alpha, \theta)$	continuous	$KC_1$	$O(K)$
SNAS	$\tilde{\mathcal{L}}(\alpha, \theta)$	continuous	$KC_1$	$O(K)$
Proxyless	$\mathcal{L}_{\theta}(z)$	BinaryConnect	$C_1 + (K - 1)C_2$	$O(1)$
ST-NAS	$\mathcal{L}(\alpha, \theta)$	ST	$C_1 + (K - 1)C_2$	$O(1)$

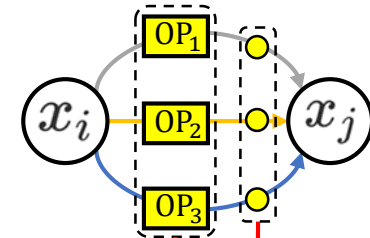
- Computation costs are estimated relative to training a single model.
- $C_1$  : the memory size for training a single model.
- $C_2$  : the average memory size for storing the output features for all connected pairs of nodes in a sub-graph. Usually we have  $C_2 \ll C_1$ .

# ST-NAS: overview

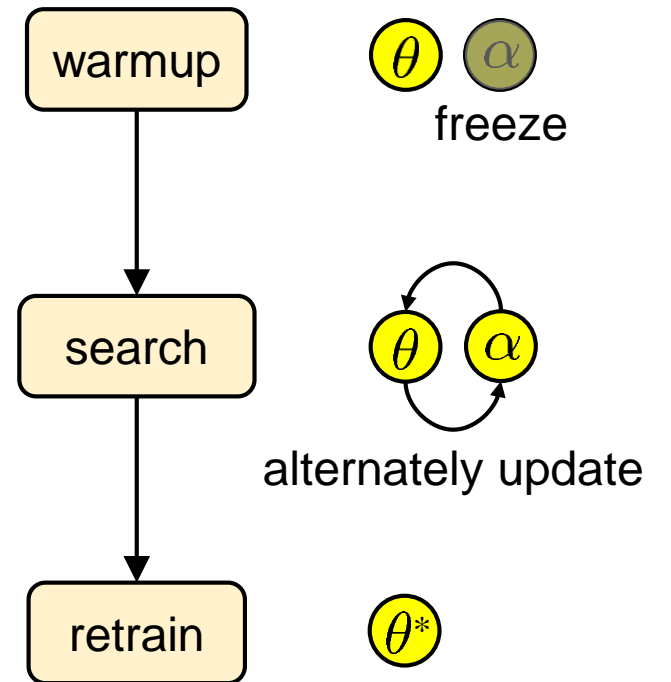
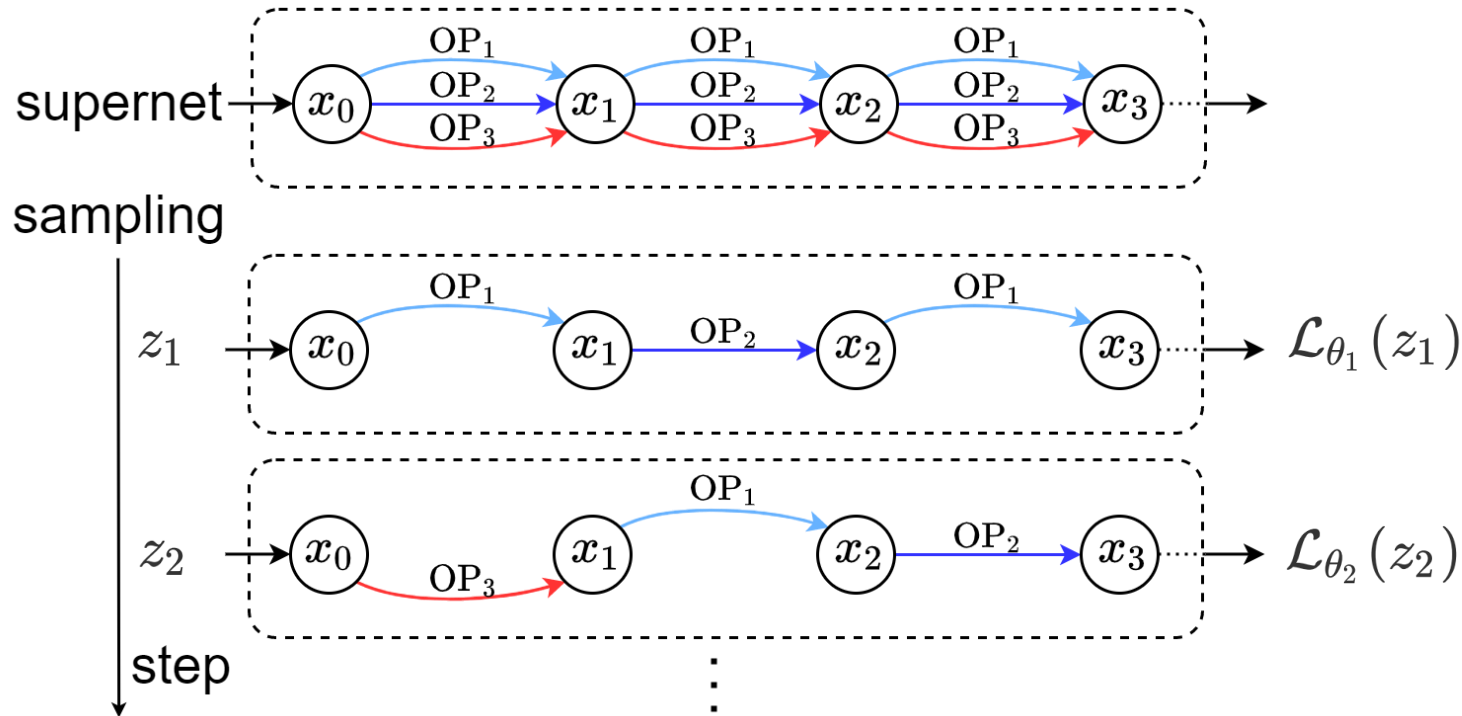
Objective:  $\mathcal{L}(\alpha, \theta) = \mathbb{E}_{z \sim p_\alpha(z)} [\mathcal{L}_\theta(z)]$

Monte Carlo estimation

sampled sub-graph



NN parameters  $\theta$   $\alpha$  architecture weights

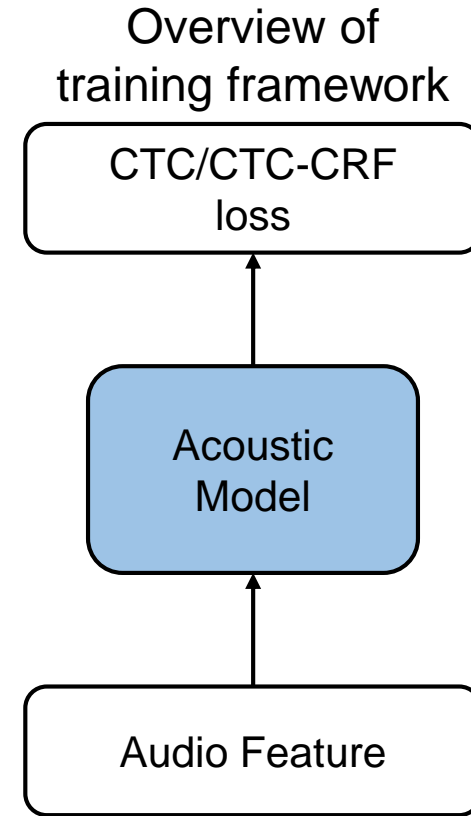
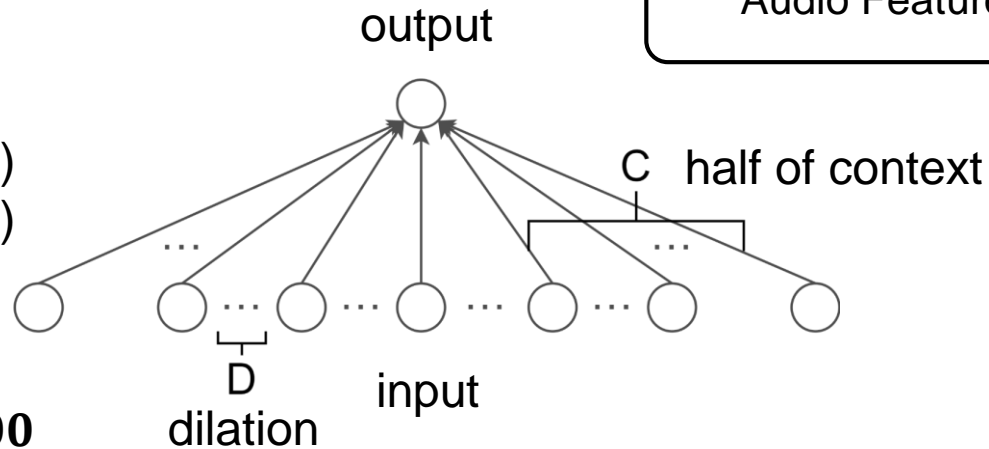


ST-NAS procedure

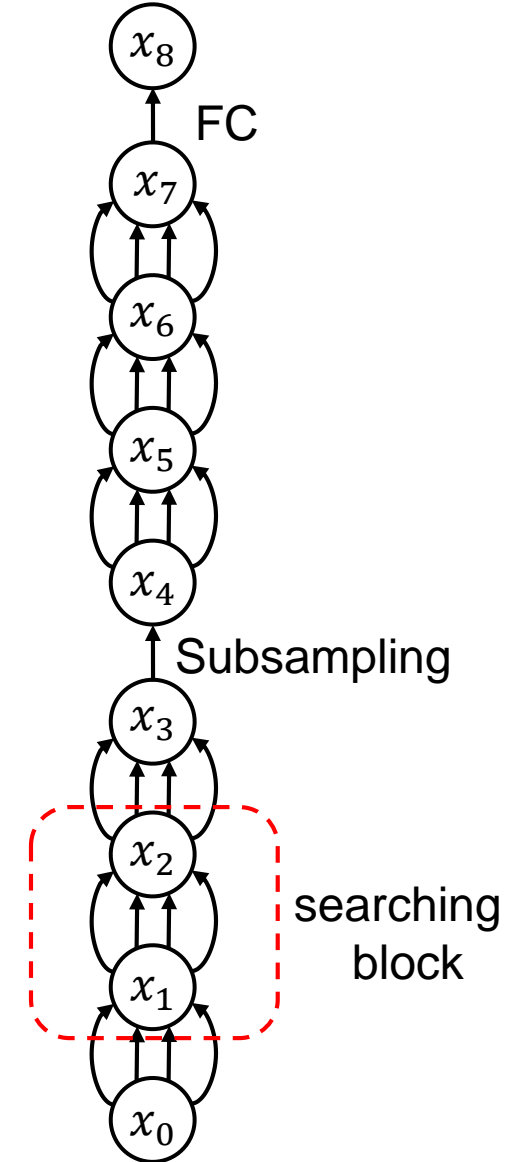
# Experiments: ASR system

Settings:

1. Datasets: 80-hour WSJ and 300-hour Switchboard.
2. LM: n-gram model.
3. Loss: CTC/CTC-CRF based on [CAT](#) [1].
4. Candidate operations:
  - TDNN-1-1 (-{C}-{D})
  - TDNN-1-2
  - TDNN-2-1
  - TDNN-2-2
  - TDNN-3-1 (Switchboard)
  - TDNN-3-2 (Switchboard)
5. Search space:
  - WSJ:  $4^6 = 4096$
  - Switchboard:  $6^6 \approx 47000$

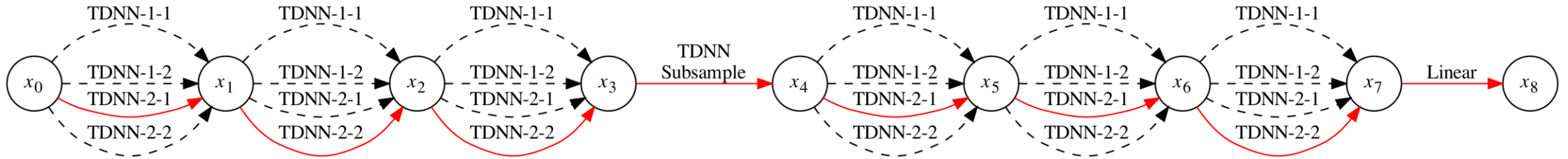


Backbone of supernet

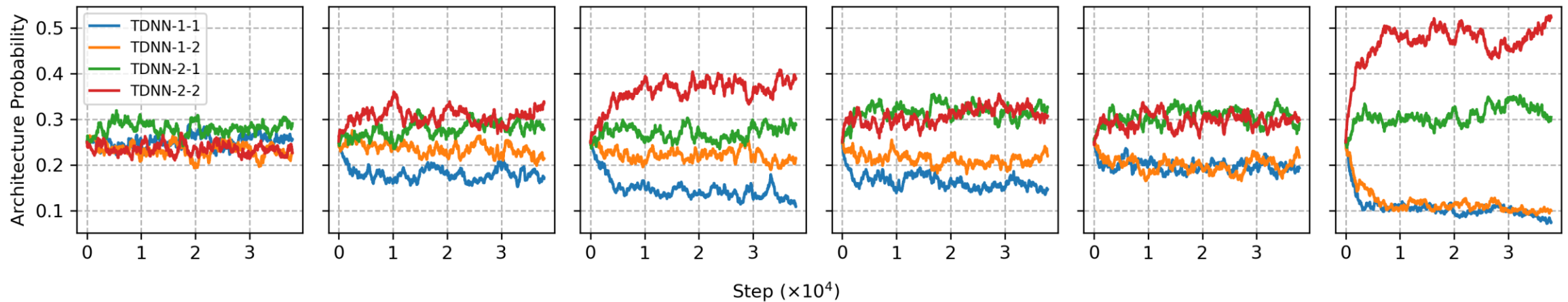


[1] K An, H Xiang, and Z Ou, "CAT: A CTC-CRF based ASR toolkit bridging the hybrid and the end-to-end approaches towards data efficiency and low latency," INTERSPEECH, 2020.

# Experiments: WSJ



The red lines indicate one of the derived single model from the 5 runs of NAS on WSJ.



The evolution of architecture probabilities for the searching blocks in the NAS run that yields the derived single model above.

# Experiments: results on WSJ

WERs on the WSJ.

Methods		eval92	dev93
EE-Policy-CTC [1]		5.53	9.21
SS-LF-MMI [2]		3.0	6.0
EE-LF-MMI [3]		3.0	-
FC-SR [4]		3.5	6.8
ESPRESSO [5]		3.4	5.9
CTC	BLSTM	4.93	8.57
	ST-NAS	4.72±0.03	8.82±0.07
CTC-CRF	BLSTM [6]	3.79	6.23
	VGG-BLSTM [7]	3.2	5.7
	TDNN-D* [8]	2.91	6.24
	Random search	2.82±0.01	5.71±0.03
	<b>ST-NAS</b>	<b>2.77±0.00</b>	<b>5.68±0.01</b>
ST-NAS with fully CTC-CRF		2.81±0.01	5.74±0.02

\* Obtained based on our implementation of the “TDNN-D” in [8].

outperforming all other end-to-end ASR models

[1] Y Zhou, C Xiong, et al, “Improving end-to-end speech recognition with policy learning,” ICASSP, 2018.

[2] H Hadian, H Sameti, et al, “Flat-start single-stage discriminatively trained HMM-Based models for ASR,” TASLP, 2018.

[3] H Hadian, H Sameti, et al, “End-to-end speech recognition using lattice-free MMI,” INTERSPEECH, 2018.

[4] N Zeghidour, Q Xu, et al, “Fully convolutional speech recognition,” arXiv preprint arXiv:1812.06864, 2018.

[5] Y Wang, T Chen, et al, “Espresso: A fast endto-end neural speech recognition toolkit,” ASRU, 2019.

[6] H Xiang and Z Ou, “CRF-based single-stage acoustic modeling with CTC topology,” ICASSP, 2019.

[7] K An, H Xiang, et al, “CAT: A CTC-CRF based ASR toolkit bridging the hybrid and the end-to-end approaches towards data efficiency and low latency,” INTERSPEECH, 2020.

[8] V Peddinti, Y Wang, et al, “Low latency acoustic modeling using temporal convolution and LSTMs,” IEEE SPL, 2018.



# Experiments: results on WSJ

WERs on the WSJ.

Methods		eval92	dev93
	EE-Policy-CTC [1]	5.53	9.21
	SS-LF-MMI [2]	3.0	6.0
	EE-LF-MMI [3]	3.0	-
	FC-SR [4]	3.5	6.8
	ESPRESSO [5]	3.4	5.9
CTC	BLSTM	4.93	8.57
	ST-NAS	4.72±0.03	8.82±0.07
	BLSTM [6]	3.79	6.23
	VGG-BLSTM [7]	3.2	5.7
CTC-CRF	TDNN-D* [8]	2.91	6.24
	Random search	2.82±0.01	5.71±0.03
	ST-NAS	<b>2.77±0.00</b>	<b>5.68±0.01</b>
	ST-NAS with fully CTC-CRF	2.81±0.01	5.74±0.02

\* Obtained based on our implementation of the “TDNN-D” in [8].

Better performance with lighter model,  
under the same CTC-CRF loss

[1] Y Zhou, C Xiong, et al, “Improving end-to-end speech recognition with policy learning,” ICASSP, 2018.

[2] H Hadian, H Sameti, et al, “Flat-start single-stage discriminatively trained HMM-Based models for ASR,” TASLP, 2018.

[3] H Hadian, H Sameti, et al, “End-to-end speech recognition using lattice-free MMI,” INTERSPEECH, 2018.

[4] N Zeghidour, Q Xu, et al, “Fully convolutional speech recognition,” arXiv preprint arXiv:1812.06864, 2018.

[5] Y Wang, T Chen, et al, “Espresso: A fast endto-end neural speech recognition toolkit,” ASRU, 2019.

[6] H Xiang and Z Ou, “CRF-based single-stage acoustic modeling with CTC topology,” ICASSP, 2019.

[7] K An, H Xiang, et al, “CAT: A CTC-CRF based ASR toolkit bridging the hybrid and the end-to-end approaches towards data efficiency and low latency,” INTERSPEECH, 2020.

[8] V Peddinti, Y Wang, et al, “Low latency acoustic modeling using temporal convolution and LSTMs,” IEEE SPL, 2018.

# Experiments: results on WSJ

WERs on the WSJ.

Methods		eval92	dev93
EE-Policy-CTC [1]		5.53	9.21
SS-LF-MMI [2]		3.0	6.0
EE-LF-MMI [3]		3.0	-
FC-SR [4]		3.5	6.8
ESPRESSO [5]		3.4	5.9
CTC	BLSTM	4.93	8.57
	ST-NAS	4.72±0.03	8.82±0.07
BLSTM [6]		3.79	6.23
VGG-BLSTM [7]		3.2	5.7
CTC-CRF	TDNN-D* [8]	2.91	6.24
	Random search	2.82±0.01	5.71±0.03
	ST-NAS	<b>2.77±0.00</b>	<b>5.68±0.01</b>
ST-NAS with fully CTC-CRF		2.81±0.01	5.74±0.02

\* Obtained based on our implementation of the “TDNN-D” in [8].

Better than strong baseline.

[1] Y Zhou, C Xiong, et al, “Improving end-to-end speech recognition with policy learning,” ICASSP, 2018.

[2] H Hadian, H Sameti, et al, “Flat-start single-stage discriminatively trained HMM-Based models for ASR,” TASLP, 2018.

[3] H Hadian, H Sameti, et al, “End-to-end speech recognition using lattice-free MMI,” INTERSPEECH, 2018.

[4] N Zeghidour, Q Xu, et al, “Fully convolutional speech recognition,” arXiv preprint arXiv:1812.06864, 2018.

[5] Y Wang, T Chen, et al, “Espresso: A fast endto-end neural speech recognition toolkit,” ASRU, 2019.

[6] H Xiang and Z Ou, “CRF-based single-stage acoustic modeling with CTC topology,” ICASSP, 2019.

[7] K An, H Xiang, et al, “CAT: A CTC-CRF based ASR toolkit bridging the hybrid and the end-to-end approaches towards data efficiency and low latency,” INTERSPEECH, 2020.

[8] V Peddinti, Y Wang, et al, “Low latency acoustic modeling using temporal convolution and LSTMs,” IEEE SPL, 2018.

# Experiments: results on WSJ

WERs on the WSJ.

Methods		eval92	dev93
	EE-Policy-CTC [1]	5.53	9.21
	SS-LF-MMI [2]	3.0	6.0
	EE-LF-MMI [3]	3.0	-
	FC-SR [4]	3.5	6.8
	ESPRESSO [5]	3.4	5.9
CTC	BLSTM	4.93	8.57
	ST-NAS	4.72±0.03	8.82±0.07
	BLSTM [6]	3.79	6.23
	VGG-BLSTM [7]	3.2	5.7
CTC-CRF	TDNN-D* [8]	2.91	6.24
	Random search	2.82±0.01	5.71±0.03
	ST-NAS	2.77±0.00	5.68±0.01
	ST-NAS with fully CTC-CRF	2.81±0.01	5.74±0.02

\* Obtained based on our implementation of the “TDNN-D” in [8].

Architectures searched with CTC are transferable to be retrained with CTC-CRF.

[1] Y Zhou, C Xiong, et al, “Improving end-to-end speech recognition with policy learning,” ICASSP, 2018.

[2] H Hadian, H Sameti, et al, “Flat-start single-stage discriminatively trained HMM-Based models for ASR,” TASLP, 2018.

[3] H Hadian, H Sameti, et al, “End-to-end speech recognition using lattice-free MMI,” INTERSPEECH, 2018.

[4] N Zeghidour, Q Xu, et al, “Fully convolutional speech recognition,” arXiv preprint arXiv:1812.06864, 2018.

[5] Y Wang, T Chen, et al, “Espresso: A fast end-to-end neural speech recognition toolkit,” ASRU, 2019.

[6] H Xiang and Z Ou, “CRF-based single-stage acoustic modeling with CTC topology,” ICASSP, 2019.

[7] K An, H Xiang, et al, “CAT: A CTC-CRF based ASR toolkit bridging the hybrid and the end-to-end approaches towards data efficiency and low latency,” INTERSPEECH, 2020.

[8] V Peddinti, Y Wang, et al, “Low latency acoustic modeling using temporal convolution and LSTMs,” IEEE SPL, 2018.

# Experiments: results on Switchboard

WERs on the Switchboard.

Methods		SW	CH	Params
	TDNN-D-Small	15.2	26.8	7.64M
	TDNN-D-Large	14.6	25.5	11.85M
ST-NAS	Transferred from WSJ	12.5	23.2	11.89M
	Searched on Switchboard	12.6	23.2	15.98M

The architecture searched in WSJ is transferable to Switchboard.

1. All experiments are trained with CTC-CRF. TDNN-D-Small is with the hidden size of 640, which is the same as that of our searched models. TDNN-D-Large is with the hidden size of 800.
2. The transferred model is randomly taken from one of the 5 runs of NAS with CTC over WSJ, and retrained on Switchboard.

# Conclusions

**NAS is an appealing next step to advancing end-to-end ASR.**

1. We review existing gradient-based NAS methods and develop an **efficient** NAS method via Straight-Through gradients (ST-NAS).
2. We **successfully** apply ST-NAS to end-to-end ASR. Our ST-NAS induced architectures significantly outperform the human-designed architecture across the WSJ and Switchboard datasets.
3. The ST-NAS method is **flexible** and can be further explored with various backbones of the supernet and candidate operations.

**Thanks for your attention!**

**Special thanks to :  
向鸿雨、安柯宇、郑华焕**

