미분가능 아키텍처 검색에서의 온도 감쇠법

동서대학교 컴퓨터공학과

석사 과정 신지용

지도교수 강대기

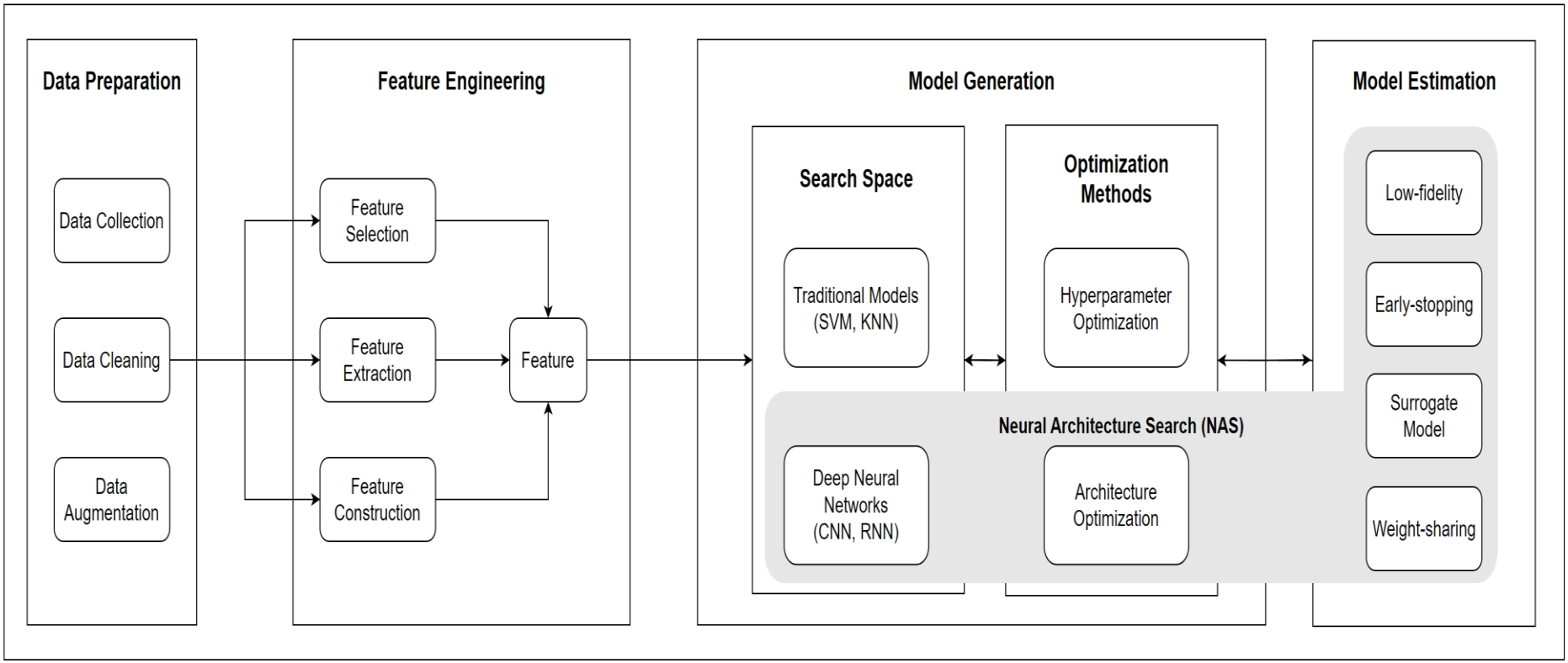
2022.11.25

목차

본 연구가 기여한 내용

- 미분 가능 아키텍처 검색(DARTS)가 가진 불일치 문제를 완화
- 아키텍처 검색시에 탐색할 양을 조절가능한 파라미터 추가
- CIFAR-10 데이터 세트에서 테스트 정확도 97.37%를 기록하여 기존의 DARTS 알고리즘 보다 0.13%p 높은 정확도

2. 자동화된 머신 러닝 (AutoML) (1/4)



- 머신 러닝 알고리즘을 개발할 시에 반복되는 작업을 자동화
- 데이터 전처리, 특징 추출, 모델 생성, 미세 조정(fine-tuning), 모델 배포 등
- 모델 설계 작업의 효율성 개선
- 모델 개발 비용 절감
- 모델 개발 시간 최소화

HPO : Hyperparameter Optimization



Hyperparameters **Parameters** **Score**


n_layers = 3
n_neurons = 512
learning_rate = 0.1

Weights optimiation
85%

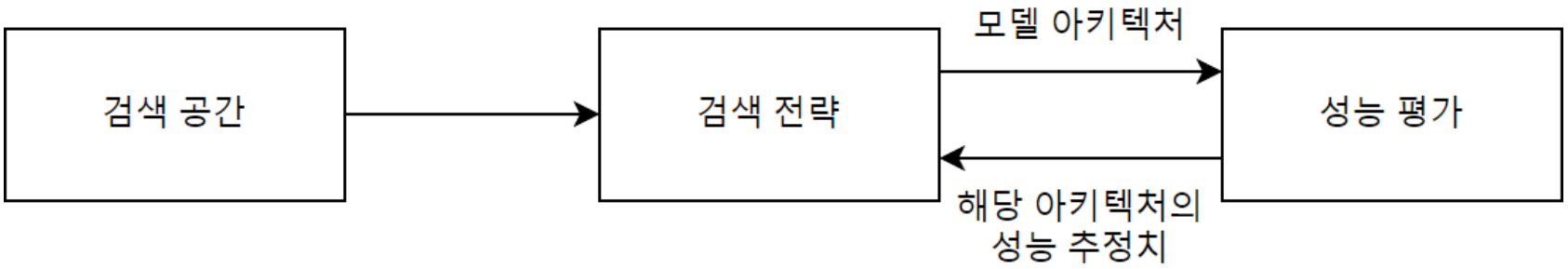

n_layers = 3
n_neurons = 1024
learning_rate = 0.01

Weights optimiation
80%


n_layers = 5
n_neurons = 256
learning_rate = 0.1

Weights optimiation
92%

NAS : Neural Architecture Search



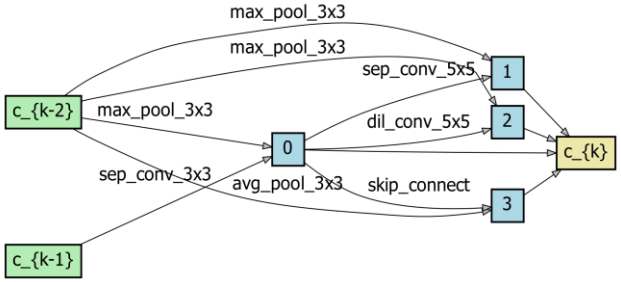
- 검색 공간을 설정
- 자신만의 검색 전략을 통해 모델 아키텍처 생성
- 해당 모델 아키텍처의 성능 평가
- 성능 평가를 통해 검색 전략을 업데이트
- 결과 아키텍처의 실제 테스트

Example: 신경망 아키텍처 검색 과정

검색 공간

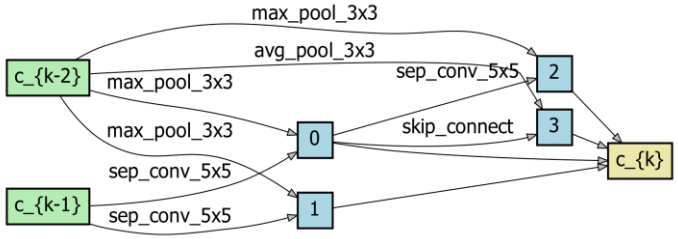
```
PRIMITIVES = [
    'none',
    'max_pool_3x3',
    'avg_pool_3x3',
    'skip_connect',
    'sep_conv_3x3',
    'sep_conv_5x5',
    'dil_conv_3x3',
    'dil_conv_5x5'
]
```

모델 아키텍처



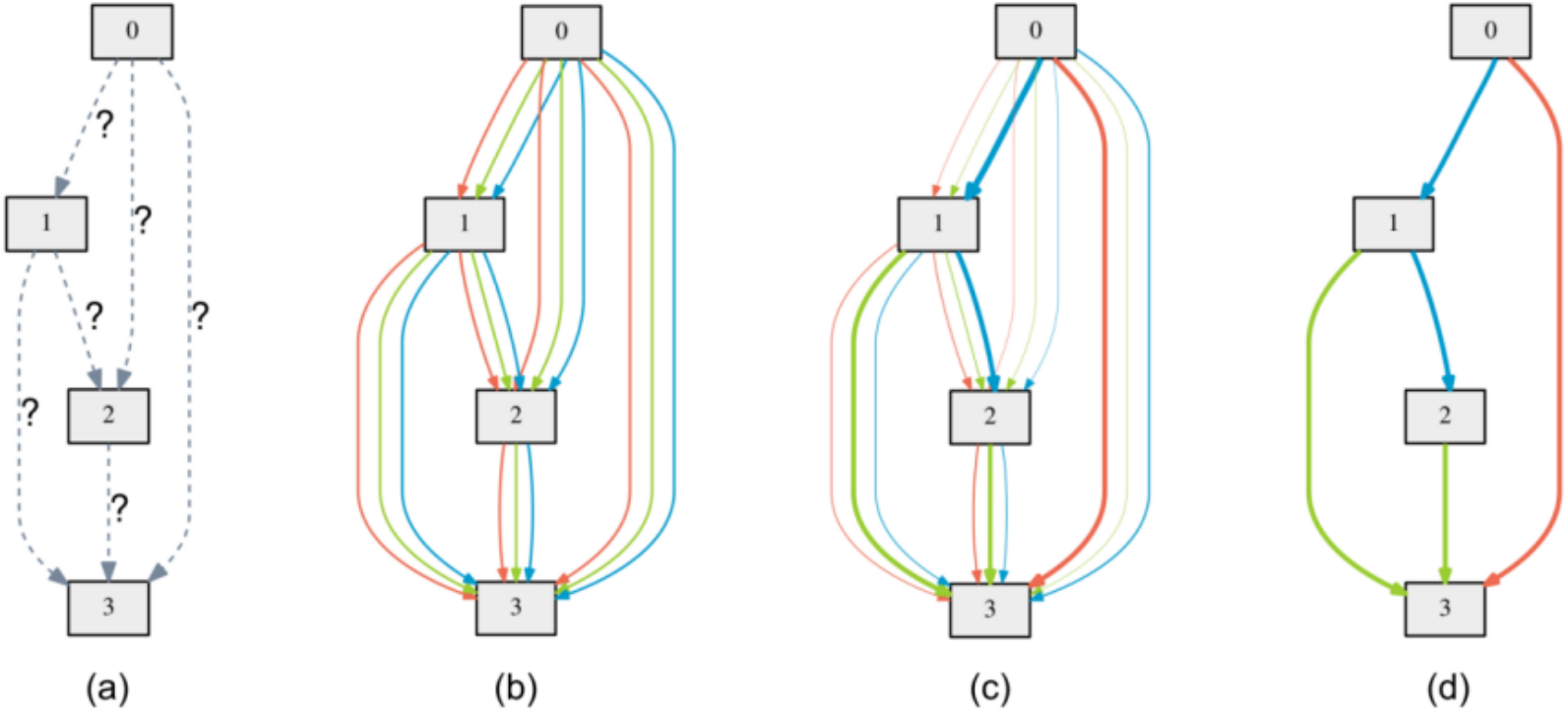
성능 평가

96.85%



96.63%

3. 미분가능 아키텍처 검색 (DARTS) (1/8)



- NAS를 유향 비순환 그래프 (DAG; Directed Acyclic Graph)로 표현
- 노드는 입출력, 간선은 연산자를 의미
- 검색은 (a) 에서 시작하여 (d) 에서 종료됨

출처: DARTS: Differentiable Architecture Search

DARTS의 검색 알고리즘

Algorithm 1: DARTS – Differentiable Architecture Search

Create a mixed operation $\bar{o}^{(i,j)}$ parametrized by $\alpha^{(i,j)}$ for each edge (i,j)

while *not converged* **do**

1. Update architecture α by descending $\nabla_\alpha \mathcal{L}_{val}(w - \xi \nabla_w \mathcal{L}_{train}(w, \alpha), \alpha)$ ($\xi = 0$ if using first-order approximation)
2. Update weights w by descending $\nabla_w \mathcal{L}_{train}(w, \alpha)$

Derive the final architecture based on the learned α.

ξ : 내부 파라미터 학습률 (0이면 1차 근사를 수행)

1. 혼합 간선(Mixed Operation)을 만든다.
2. 내·외부 파라미터를 학습시킨다.
3. 학습된 아키텍처 파라미터(외부 파라미터)로 아키텍처를 생성한다.
4. 생성된 아키텍처로 내부 파라미터를 초기화 시키고 재학습한다.

출처: DARTS: Differentiable Architecture Search

DARTS의 근사법

1)
$$\nabla_\alpha \mathcal{L}_{val}(w^*(\alpha), \alpha)$$
$$\approx \nabla_\alpha \mathcal{L}_{val}(w - \xi \nabla_w \mathcal{L}_{train}(w, \alpha), \alpha)$$

2)
$$\nabla_\alpha \mathcal{L}_{val}(w', \alpha) - \xi \nabla^2_{\alpha,w} \mathcal{L}_{train}(w, \alpha) \nabla_{w'} \mathcal{L}_{val}(w', \alpha)$$

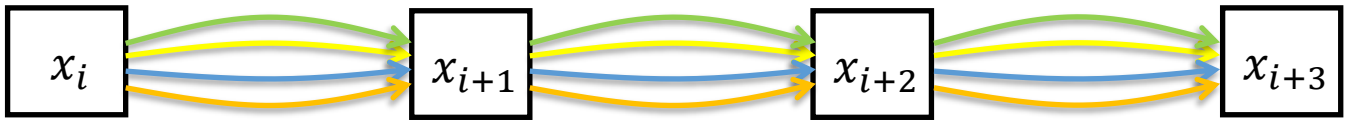
3)
$$\nabla^2_{\alpha,w} \mathcal{L}_{train}(w, \alpha) \nabla_{w'} \mathcal{L}_{val}(w', \alpha) \approx \frac{\nabla_\alpha \mathcal{L}_{train}(w^+, \alpha) - \nabla_\alpha \mathcal{L}_{train}(w^-, \alpha)}{2\epsilon}$$

- 내부 파라미터를 업데이트 한 후 외부 파라미터 업데이트 (양단식 최적화)
- 식 2)와 3)은 알고리즘에 소개된 식 1)로 인해 유도된 식
- 양단식 최적화를 유한 차분법을 통해 근사
- 라플라시안 연산자로 인한 지연을 델 연산자로 근사하여 시간을 단축시킴

출처: DARTS: Differentiable Architecture Search

3. 미분가능 아키텍처 검색 (DARTS) (4/8)

Operation pool = [Convolution,
Pooling,
Skip,
Dilation]



$$\bar{o}^{(i,j)}(x_i) = \sum_{o \in \mathcal{O}} \frac{\exp\left(\alpha_o^{(i,j)}\right)}{\sum_{o' \in \mathcal{O}} \exp\left(\alpha_{o'}^{(i,j)}\right)} o(x_i) \;=\; x_{i+1}$$

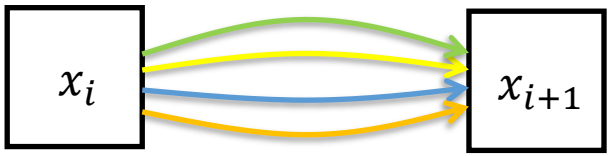
i : i 번째 간선
j : i 번째 간선의 j 번째 연산자

$o \in \mathcal{O}$: 검색할 연산자 집합 O에서 선택된 k번째 연산자

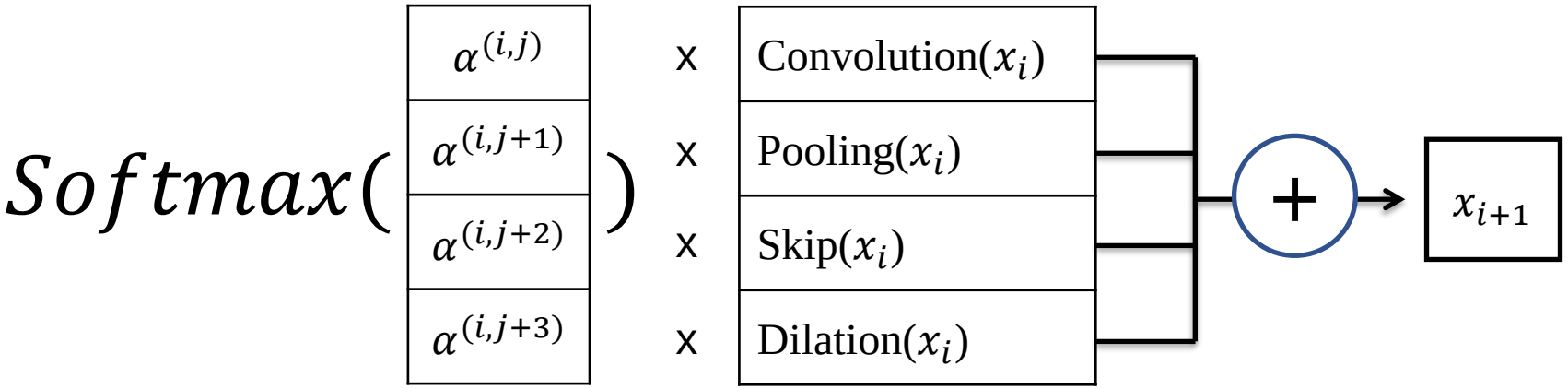
$\alpha^{(i,j)}$: 아키텍처 파라미터 (벡터)

$\bar{o}^{(i,j)}$: 혼합 연산자 (Mixed operation)

3. 미분가능 아키텍처 검색 (DARTS) (5/8)

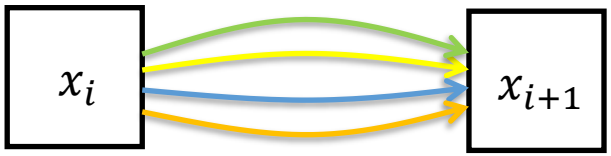


$$\bar{o}^{(i,j)}(x_i) = \sum_{o \in \mathcal{O}} \frac{\exp\left(\alpha_o^{(i,j)}\right)}{\sum_{o' \in \mathcal{O}} \exp\left(\alpha_{o'}^{(i,j)}\right)} o(x_i)$$

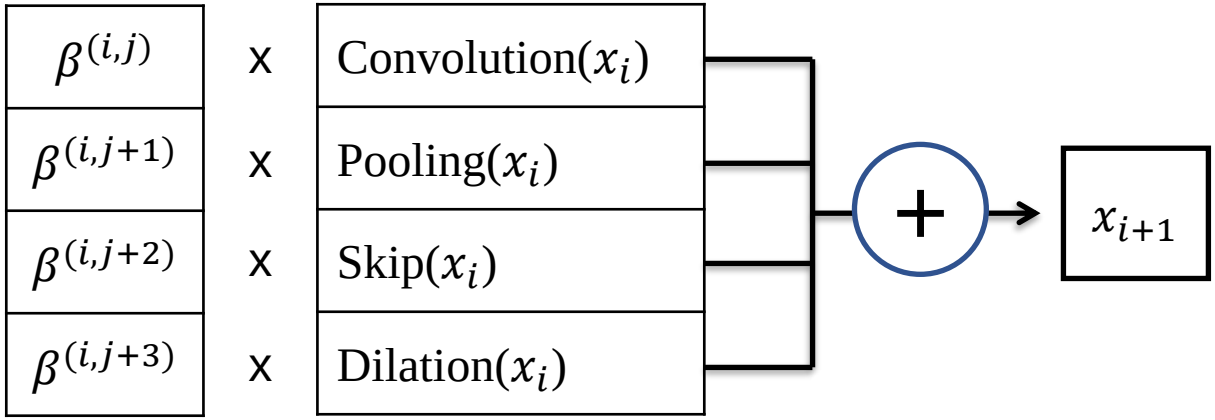


$\alpha \cdot \text{Convolution}(x_i) + \alpha \cdot \text{Pooling}(x_i) + \alpha \cdot \text{Skip}(x_i) + \alpha \cdot \text{Dilation}(x_i) = \text{out}$

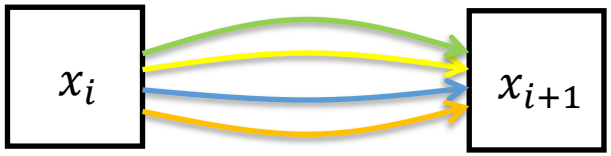
3. 미분가능 아키텍처 검색 (DARTS) (6/8)



$$\bar{O}^{(i,j)}(x) = \sum_{k=1}^{|\mathcal{O}|} \beta_k^{(i,j)} O_k(x)$$



$\beta \cdot$ Convolution(x_i) + $\beta \cdot$ Pooling(x_i) + $\beta \cdot$ Skip(x_i) + $\beta \cdot$ Dilation(x_i) = out



Operation pool = [Convolution,
Pooling,
Skip,
Dilation]

$$\beta_k^{(i,j)} = \frac{\exp\left(\alpha_o^{(i,j)}\right)}{\sum_{o' \in \mathcal{O}} \exp\left(\alpha_{o'}^{(i,j)}\right)}$$

$$\bar{O}^{(i,j)}(x) = \sum_{k=1}^{|\mathcal{O}|} \beta_k^{(i,j)} O_k(x)$$

$k \in \mathcal{O}$: 검색할 연산자 집합 O에서 선택된 k번째 연산자

$\alpha^{(i,j)}$: 아키텍처 파라미터 (벡터)

$\bar{o}^{(i,j)}$: 혼합 연산자 (Mixed operation)

3. 미분가능 아키텍처 검색 (DARTS) (8/8)

4 Conclusion

We presented DARTS, a simple yet efficient architecture search algorithm for both convolutional and recurrent networks. By searching in a continuous space, DARTS is able to match or outperform the state-of-the-art non-differentiable architecture search methods on image classification and language modeling tasks with remarkable efficiency improvement by several orders of magnitude.

There are many interesting directions to improve DARTS further. For example, the current method may suffer from discrepancies between the continuous architecture encoding and the derived discrete architecture. This could be alleviated, e.g., by annealing the softmax temperature (with a suitable schedule) to enforce one-hot selection. It would also be interesting to investigate performance-aware architecture derivation schemes based on the shared parameters learned during the search process.

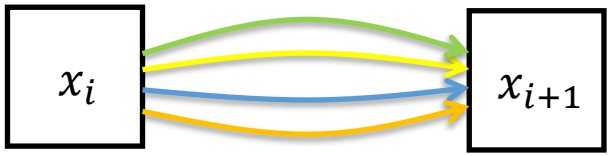
- **문제점**
  - 불일치 : 연속적으로 인코딩된 아키텍처와 실제 아키텍처의 이산성 간의 불일치
- **해결 방법**
  - 소프트맥스 함수의 온도를 조절함으로써 완화

출처: DARTS: Differentiable Architecture Search

4. 온도 감쇠 미분가능 아키텍처 검색(TD-DARTS) (1/7)

$\beta^{(i,j)}$	x	Convolution(x_i)		0.2		1	0	0	0
$\beta^{(i,j+1)}$	x	Pooling(x_i)		0.25	≠	0	1	0	0
$\beta^{(i,j+2)}$	x	Skip(x_i)		0.4		0	0	1	0
$\beta^{(i,j+3)}$	x	Dilation(x_i)		0.15		0	0	0	1

- **가정**
  - 최종 선택과 DARTS의 weighted sum 간에 불일치 발생
- **해결 방법**
  - 소프트맥스 함수의 입력값을 조정 (exploration/exploitation)

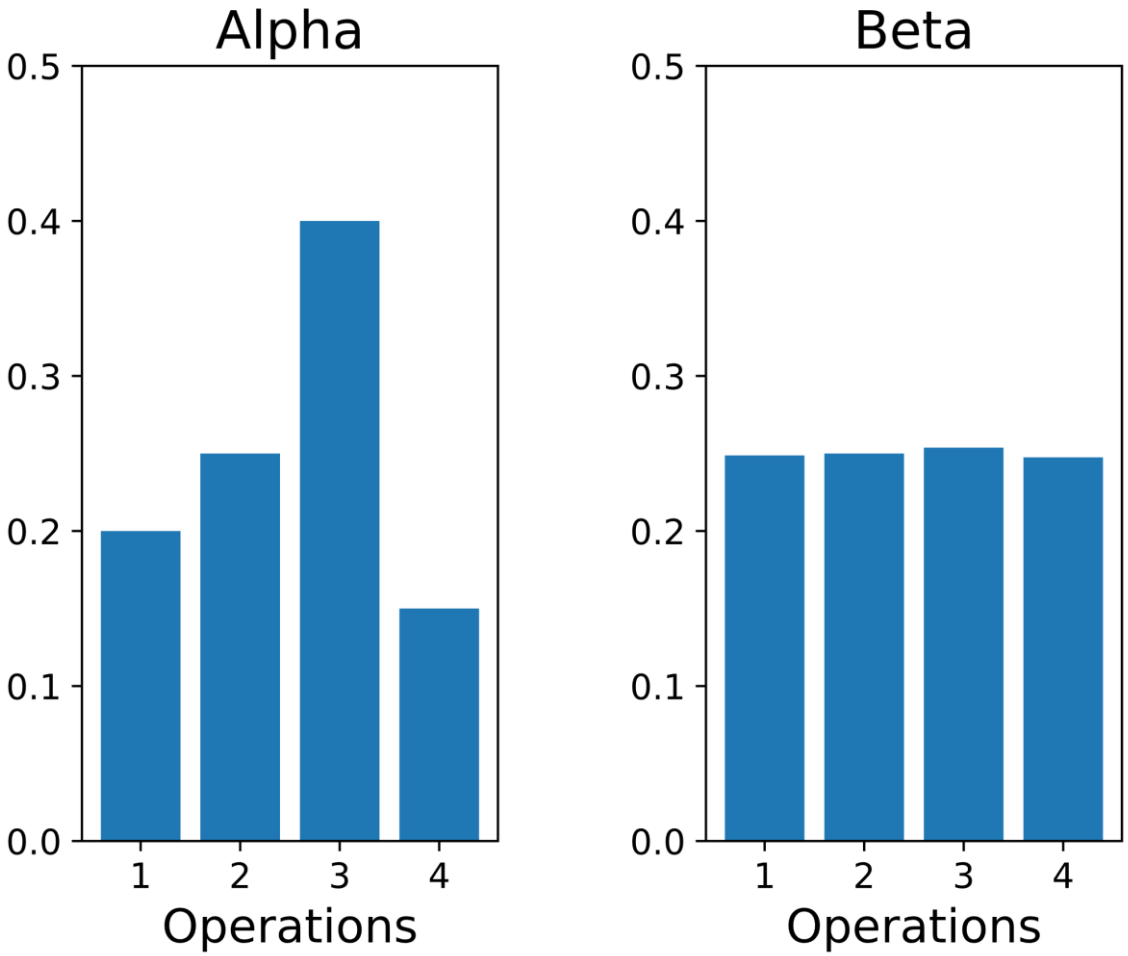


Operation pool = [Convolution,
Pooling,
Skip,
Dilation]

$$\beta_k^{(i,j)} = \frac{exp(\alpha_k^{(i,j)}/T)}{\sum_{k'}^{O} exp(\alpha_{k'}^{(i,j)}/T)}$$

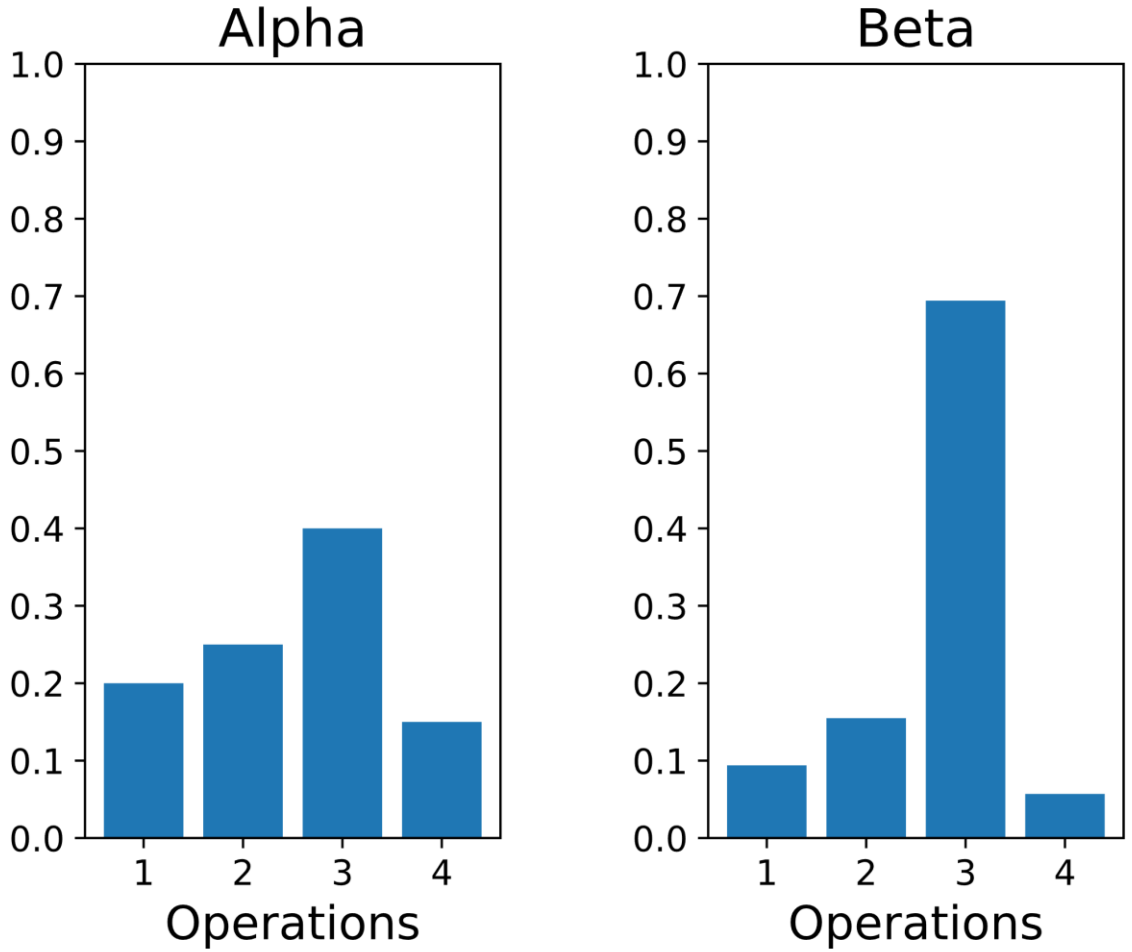
- 소프트맥스 함수에 온도를 적용
- 아키텍처 파라미터를 리스케일링 (Rescaling)
- β 값의 조절
- 연산자들의 기여도 조절
- 검색하는 연산자들의 경쟁 심화 상태에서 완화 상태로 변화

- 온도 T 값이 높을 때



- 높은 exploration, 낮은 exploitation

- 온도 T 값이 낮을 때



- 높은 exploitation, 낮은 exploration

$$\beta_k^{(i,j)} = \frac{exp(\alpha_k^{(i,j)}/T)}{\sum_{k'}^{O} exp(\alpha_{k'}^{(i,j)}/T)}$$

$$T = temp_{init} - \frac{temp_{init} - temp_{end}}{epoch - 1}$$

ex)

$temp_{init}$ = 10

$temp_{end}$ = 0.1

$\frac{10-0.1}{50-1} = \frac{9.9}{49} = 0.20204 \ldots$

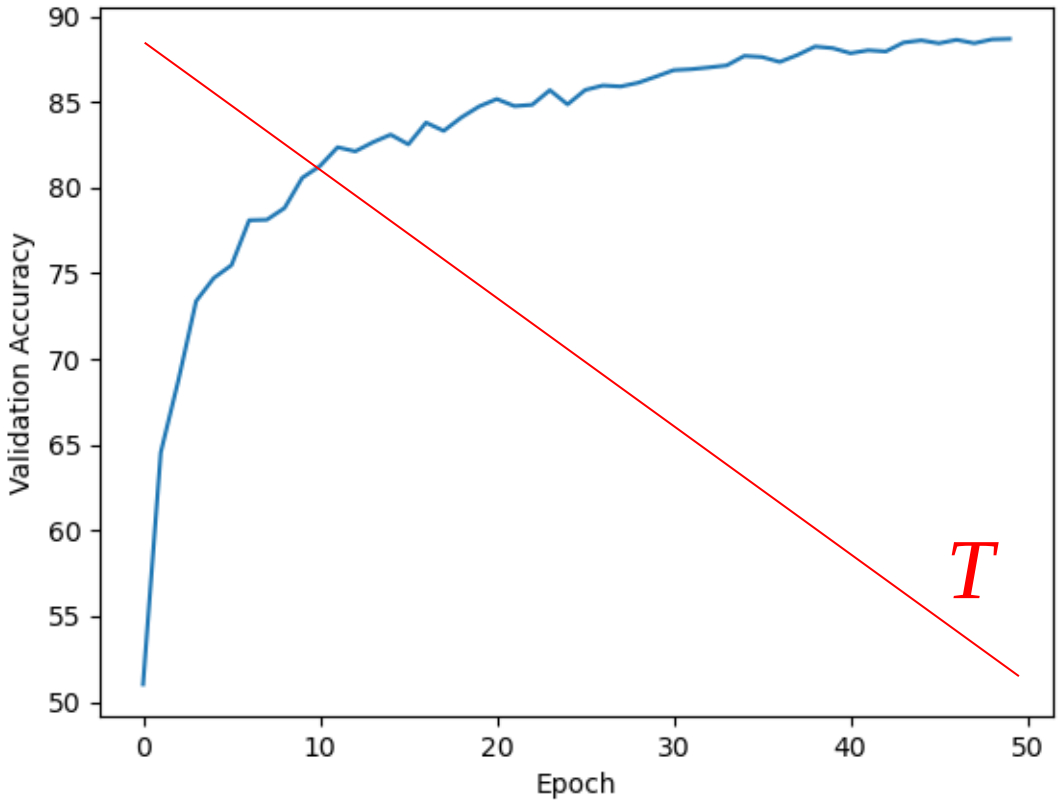
epoch0 : T = 10

epoch1 : T ≈ 9.8

⋮

epoch49 : T = 0.1

4. 온도 감쇠 미분가능 아키텍처 검색(TD-DARTS) (6/7)



- 온도를 선형적으로 감쇠
- 초기 온도는 10, 최종 온도는 0.1로 설정
- 매 에포크마다 약 0.20 정도의 온도가 감쇠
- 검색 초기에는 연산자간의 경쟁을 강화하여 탐색 위주의 검색을 하고, 검색 후기에는 이산적인 벡터가 되도록 함

- Optimization function of DARTS

$$\min_{\alpha} \mathcal{L}_{val}\left(w^*(\alpha), \alpha\right)$$

$$\text{s.t. } w^*(\alpha) = \arg\min_{w} \mathcal{L}_{train}(w, \alpha)$$

- How the alpha updated

$$\alpha_k^{t+1} \leftarrow \alpha_k^t - \eta_\alpha \cdot \nabla_{\alpha_k} \mathcal{L}_{val}$$

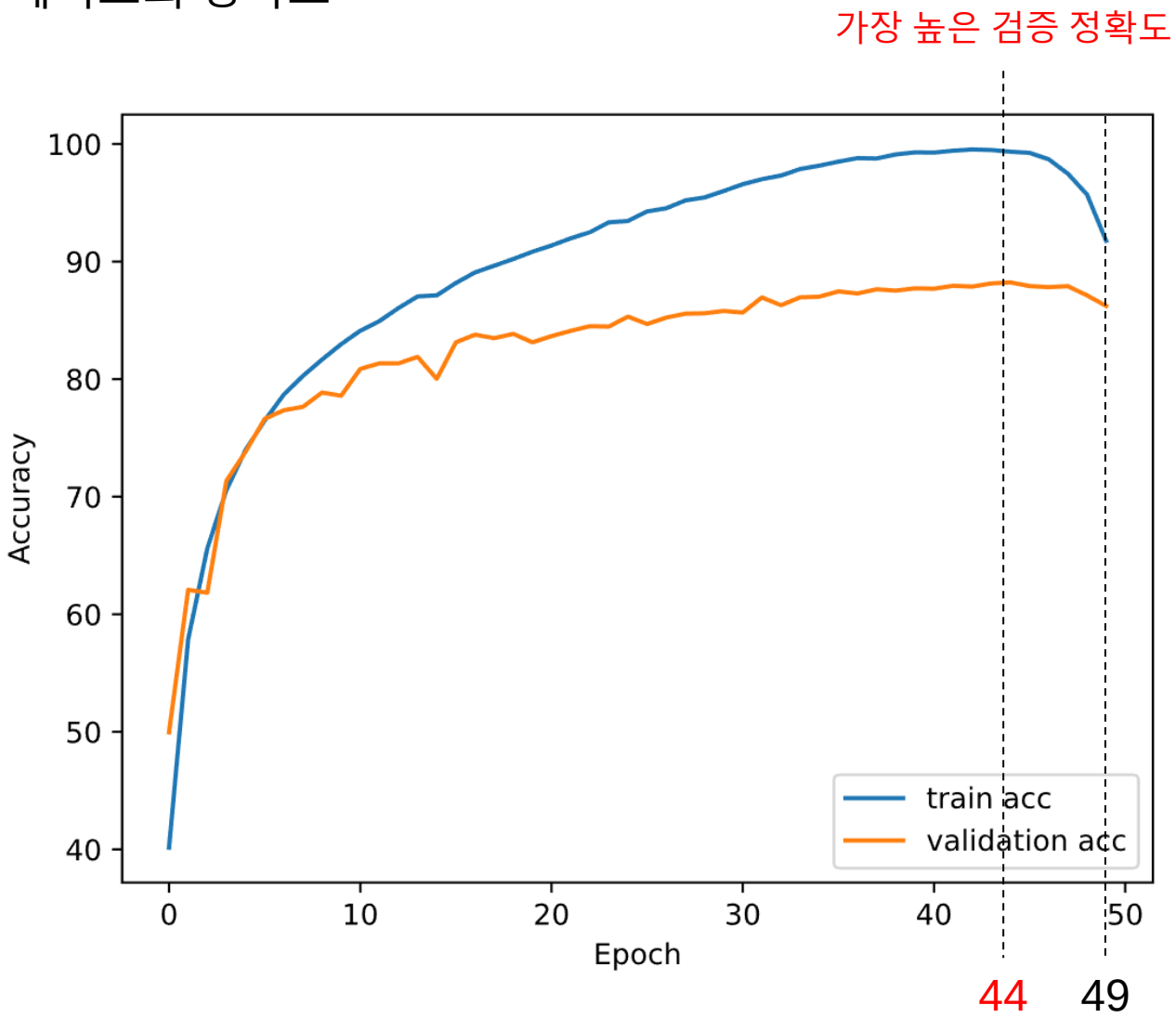
k : 간선 번호

- Temperature Annealing

$$\alpha_k^{t+1} \leftarrow \frac{\alpha_k^t - \eta_\alpha \nabla_{\alpha_k} \mathcal{L}_{val}}{T^{t+1}}$$

5. 실험 결과 (1/4)

- 검색 페이즈의 정확도



Genotypes

- Epoch 44

```
Normal cell=
[('sep_conv_3x3', 1),
('sep_conv_5x5', 0),
('skip_connect', 0),
('sep_conv_3x3', 1),
('sep_conv_3x3', 1),
('skip_connect', 0),
('skip_connect', 0),
('sep_conv_3x3', 2)]

Reduce cell=
[('max_pool_3x3', 0),
('max_pool_3x3', 1),
('max_pool_3x3', 0),
('dil_conv_3x3', 1),
('max_pool_3x3', 0),
('skip_connect', 3),
('skip_connect', 3),
('avg_pool_3x3', 0)]
```

- Epoch 50

```
Normal cell=
[('skip_connect', 0),
('sep_conv_3x3', 1),
('skip_connect', 1),
('sep_conv_5x5', 0),
('skip_connect', 2),
('max_pool_3x3', 1),
('max_pool_3x3', 0),
('max_pool_3x3', 1)],

Reduce cell=
[('sep_conv_5x5', 0),
('sep_conv_5x5', 1),
('skip_connect', 2),
('avg_pool_3x3', 0),
('skip_connect', 2),
('sep_conv_3x3', 1),
('skip_connect', 2),
('skip_connect', 3)]
```

5. 실험 결과 (3/4)

Epoch	Val acc (%)	Train acc (%)	Total # of skip connection
40	87.692	99.272	5
41	87.940	99.428	5
42	87.872	99.536	5
43	88.140	99.488	5
44	88.228	99.344	5
45	87.916	99.248	6
46	87.824	98.700	6
47	87.908	97.476	8
48	87.124	95.708	8
49	86.224	91.776	8
50	-	-	7

CIFAR-10 데이터 세트에서 검색 및 훈련 결과

Genotype	Test acc (%)
DARTS_V2	97.24
TD-DARTS_50	96.94
TD-DARTS_44	97.37

- DARTS_V2 모델은 2차 근사 모델
- 50번째 에포크의 모델보다 44번째 모델의 정확도가 더 높으며, 이는 기존 DARTS_V2 모델보다 0.13%p 더 높은 정확도

6. 결론

1. DARTS의 불일치 문제 완화
   - 연속적으로 인코딩된 아키텍처와 최종 아키텍처간의 불일치 문제
   - 논문에서는 소프트맥스 함수에 온도를 적용함으로써 완화가 가능할 것이라 예상 ("This could be alleviated by annealing the softmax temperature (with a suitable schedule) to enforce one-hot selection") [DARTS]
   - 온도를 작은 값으로 조절함으로써 불일치 문제를 완화

2. Exploration 및 Exploitation 조절
   - 온도를 10에서 0.1로 낮춤으로써 검색 초기에 더 풍부한 검색을 하도록 설정하였음

3. 개선 사항 및 향후 연구 방향
   - DARTS를 표방한 다른 알고리즘에 적용가능할 것
   - 선형 감쇠가 아닌 다른 방법으로 접근 가능

참고 문헌

- **DARTS: Differentiable Architecture Search** https://arxiv.org/abs/1806.09055
- **Differentiable Architecture Search with Ensemble Gumbel-Softmax** https://arxiv.org/abs/1905.01786
- **Categorical Reparameterization with Gumbel-Softmax** https://arxiv.org/abs/1611.01144
- **Fair DARTS: Eliminating Unfair Advantages in Differentiable Architecture Search** https://arxiv.org/abs/1911.12126
- **β-DARTS: Beta-Decay Regularization for Differentiable Architecture Search** https://arxiv.org/abs/2203.01665
- **AutoML: A Survey of the State-of-the-Art** https://arxiv.org/abs/1908.00709
- **Neural Architecture Search: A Survey** https://arxiv.org/abs/1808.05377

감사합니다

Q & A