

Nas net. → RL을 사용해 architecture를 최적화하는 framework
Nasnet search space. → 새로운 Space search를 디자인 함으로써, transfer ability를 확장함.

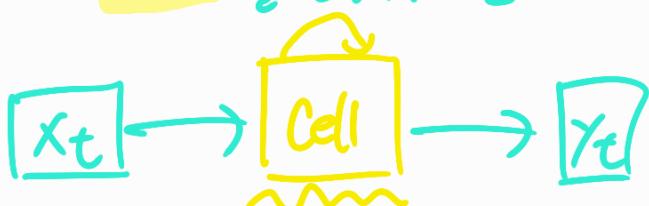
모든 Convolution network는 'weight 값을 다룬
구조는 동일한 Convolution 또는 Cell을'로 이루어짐.

↳ 최적의 Convolution 찾는 문제 → 최적의 Cell 구조
 찾는 문제.

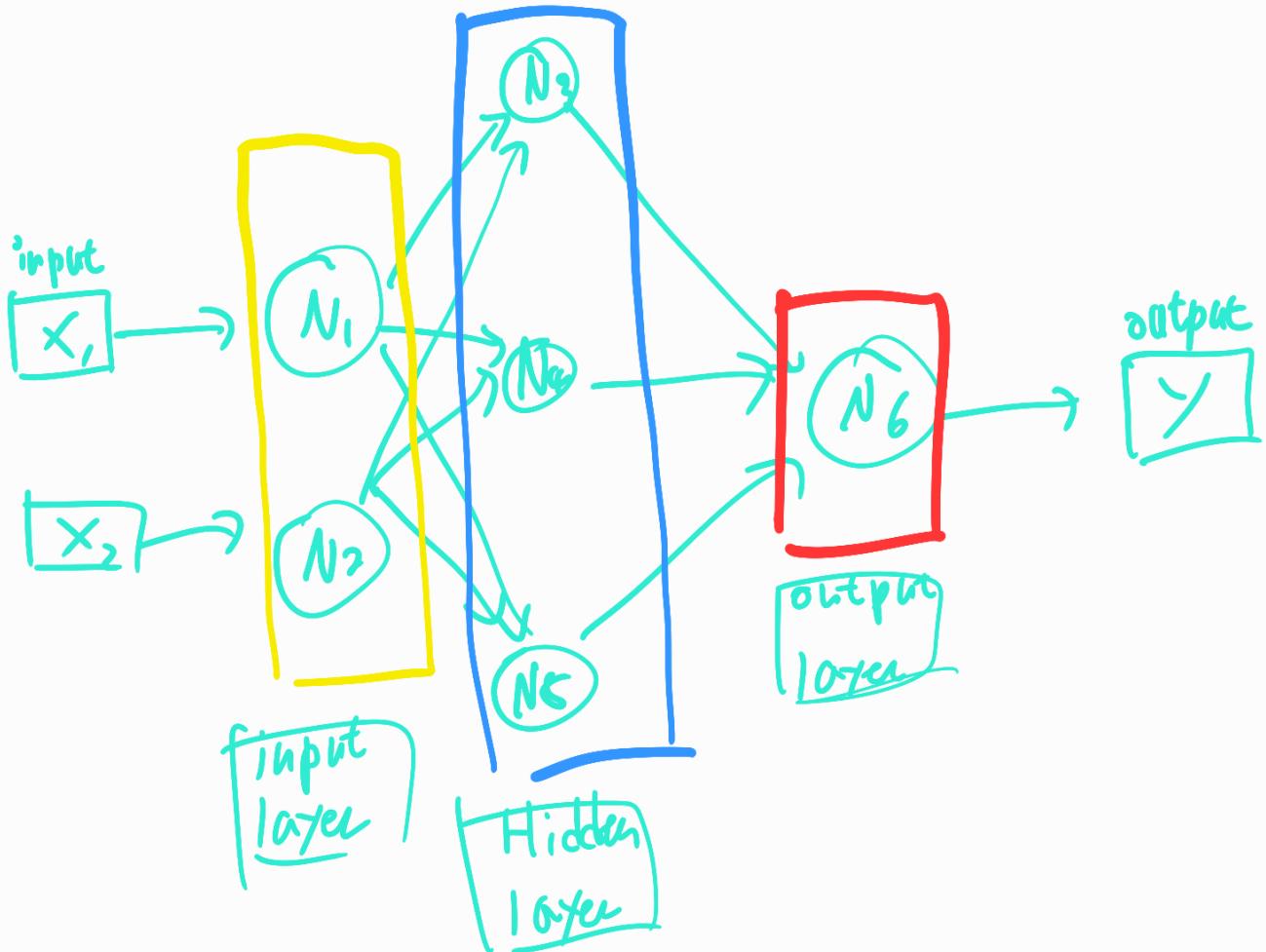
- ↳ 1) 전체 네트워크 찾는 것보다 fast,
 2) Cell 자체는 다른 문제에 의해 더 잘 풀릴 수 있는
 될 수 있다.

* Cell과 layer.

○ Cell: RNN에서 은닉층에서 활성화 함수를 통해 결과를 내보내는 역할을 하는
 node를 Cell이라고 한다.



○ layer: 입력/은닉/출력 층과 같은 ④



* search space

Search space는 알고리즘이 검색하는 집합(set) 또는 도메인.

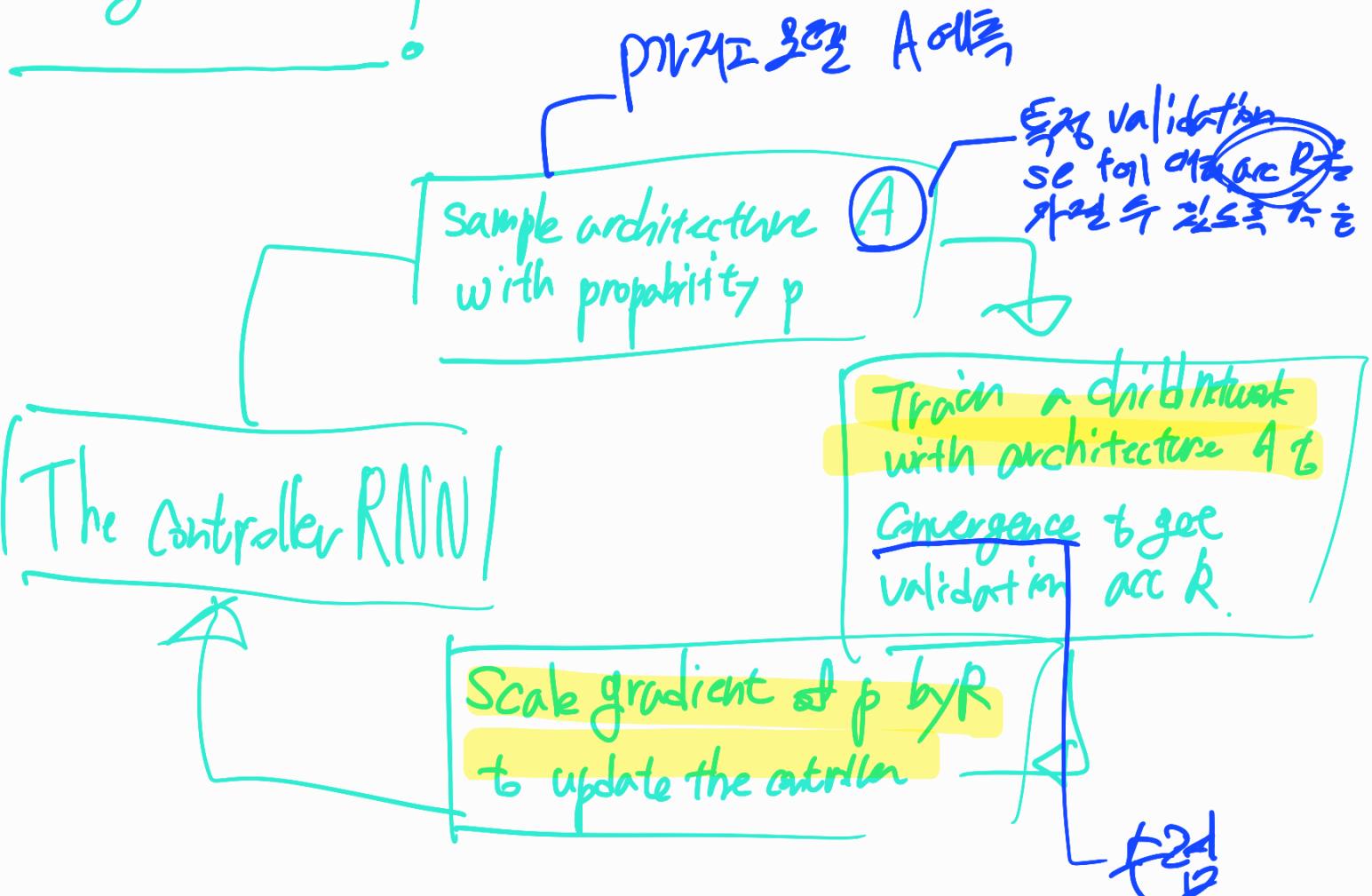
In computer science

↳ Space = 잘 정의되고 유한한 data structure.

e.g.) 체스

체스를 보면 터미널환경에서 있으나, 플레이어가 행동하는 경지의 수는 무한하다 → 이걸 확률이 최대 가 되도록 한다.

using NasNet.]



<NAS framework>

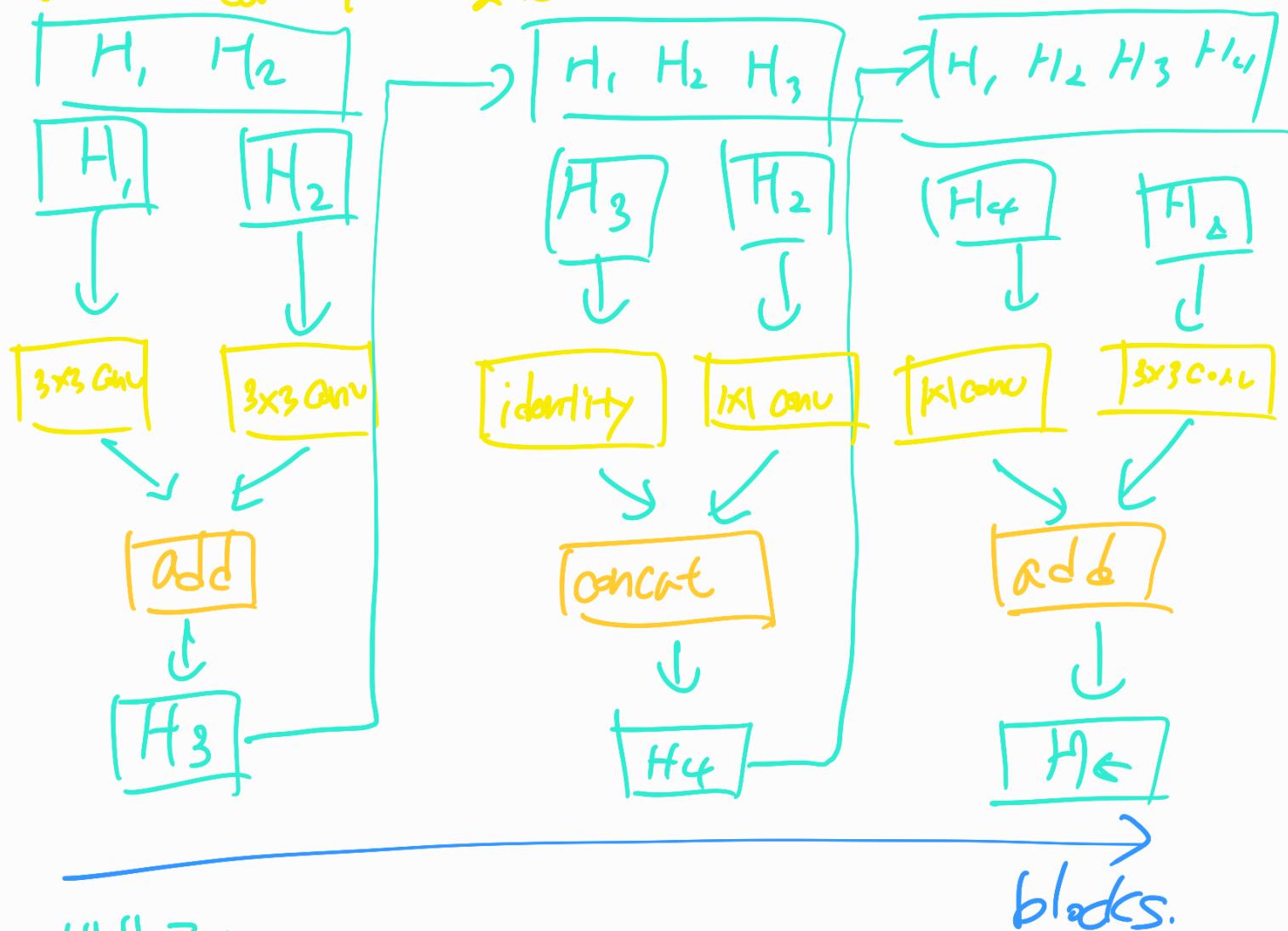
[NAS Net search space]

Conditional network 구조는 미리 수동으로 결정

image size와 함께 확장 가능한 구조를 만들기 위해 feature map을 input으로 가져올 때 두 가지 주요한 일을 수행하는 두 가지 종류의 convolutional cell이 필요.

- 1) Normal cell: 같은 차원의 feature map을 합친다는 convolution.
- 2) Reduction cell: 높이와 너비가 1/2인 feature map을 합친다는 convolution.
cells - 차원에 cell의 input을 stride 2로 헤쳐온다

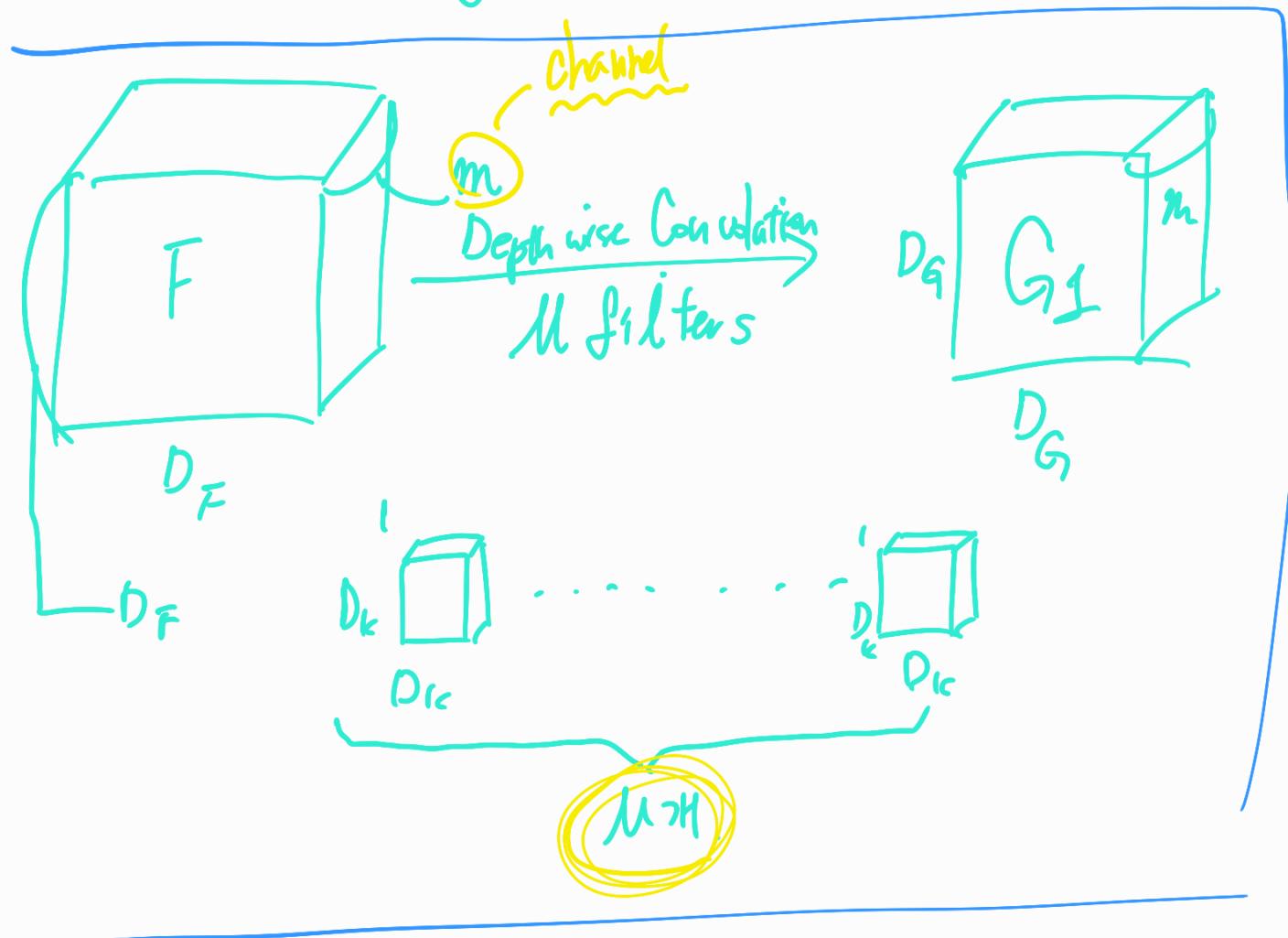
Controller RNN Non-ejan NASNet Search Space의 Normal & Reduction Cell 구조를 찾았다.



* 세부구조

- 모든 convolution은 ReLU를 적용
- 필터에 따라 1×1 convolution을 사용해 shape를 맞춘다.
- 모든 Depth wise separable convolution은 depth wise의 pointwise 단위에서 BN과 ReLU를 둘 다 적용 X. (or, not)

⇒ depth wise convolution
: filtering stage

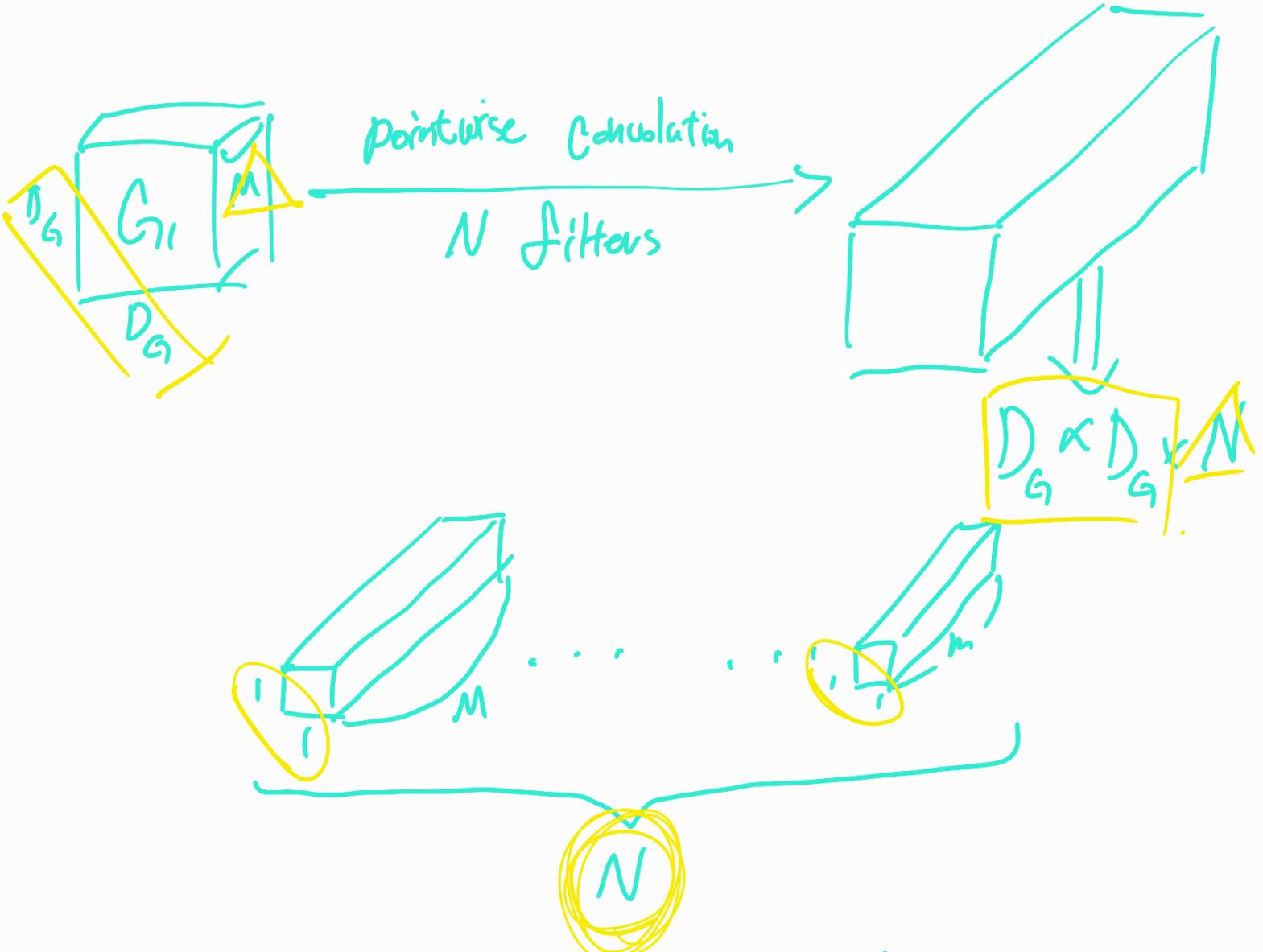


⇒ Standard Convolution → **1개의 filter가 M 차수로 전개하여 convolution**

연산

{ But 한개의 filter는 1개의 차수로 예전 연산, input과 같은 차수로 학습하는 경우 $(D_F, D_F, 1)$ 이라 하면, 최근에는 연산되는 filter $(D_G, D_G, 1)$ 이 존재.

▷ pointwise convolution.



- 1×1 convolution 적용. (D_G, D_G, M) 의 $(1, 1, M)$ 에 $conv$ 연산.

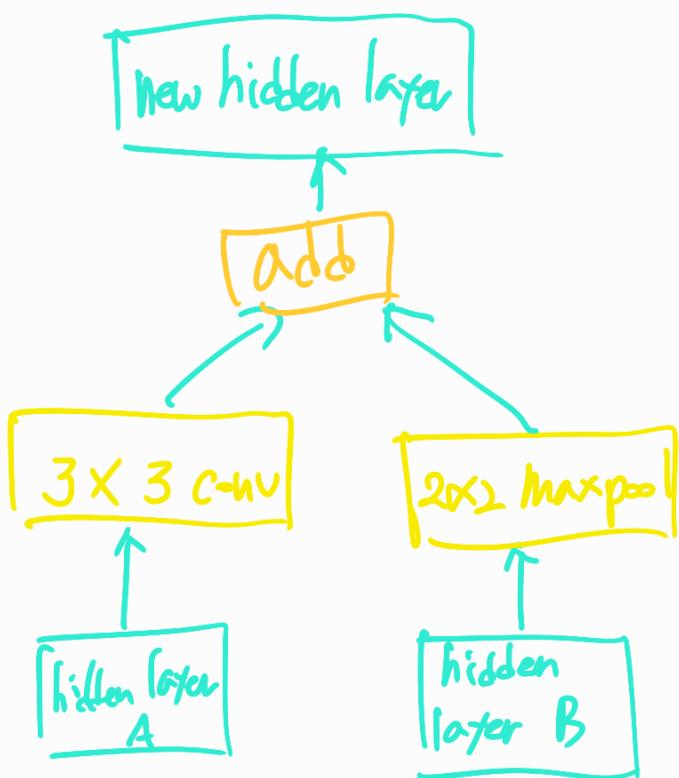
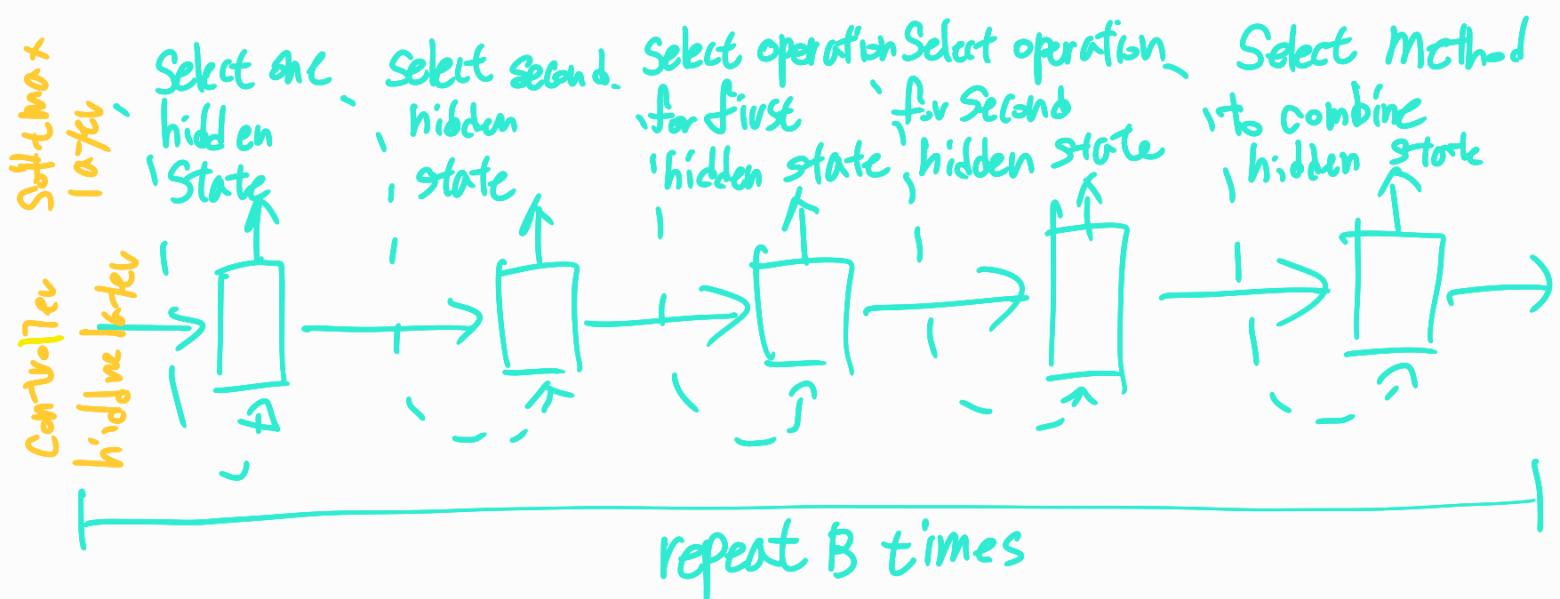
- $(D_G, D_G, M) \otimes (1, 1, M) = \underbrace{(D_G, D_G, 1)}_{\times N \text{ 개 필터}}$ 이 된다.

• 모든 convolution은 identity mapping이 $ReLU$, $conv$ 연산, BN 순서를 따름

✓ separable convolution이 영상으로 처리되는면, 이는 hidden state or depth에 영향을 미친다. (성능 향상 가능)

→ depthwise Separable Convolution.

* RNN Controller (1 layer LSTM with 100 hidden units)



- 6개의 parameter를 가지는 softmax layer를 가진다.
- 하나의 cell은 block B 개를 갖기 때문에 controller는 convolutional cell 구조를 예측하기 위해 $6B$ 개의 softmax layer를 갖고 있다.

□ Controller Step

- ▷ h_i, h_{i+1} 또는 이전 block의 hidden state로 부터 hidden state를 고운다.
- ▷ step과 같은 보통의 선택된 hidden state에 적용될 operation을 고른다.
- ▷ Step 1에서 선택된 hidden state에 적용될 operation을 고운다.

4) Step 2에서 11

5) 시도운 hidden state를 만들기 위해 step 3, 4의 결과를 합을
방법을 고른다. (element-wise addition or concatenation between
two hidden states)

ⓐ 초기화로 RNN의 Normal cell이 Reduction cell을 둘 때 학습할 수
있게 $2 \times 6B$ 의 예측을 하도록 함
 $6B + 6B = 2 \times 6B$.

□ element-add

layer A ($2, 1, 4, 2$) layer B ($0, 1, 0, 1$) - : \oplus
layer A + layer B = $(2, 2, 4, 3)$

□ concat

