요약

- ST-GCN 문제점을 보완하기 위해 나온 논문
  - ST-GCN의 경우 local 영역에서의 관절간의 관계성 밖에 찾지 못함
  - ST-GCN의 경우 topology가 고정되어있기 때문에 기존의 GCN이 갖고 있는 계층적인 특성을 갖지 못함
  - 고정된 그래프 구조는 모든 sample들에게 최적화 되어있지 않음 (ex 얼굴을 쓸어내리는 동작의 경우 손과 머리의 관계성이 중요한데에 비해 점프하는 동작은 필요없음)
  - joint position 뿐만 아니라 joint와 joint를 잇는 bone 정보(길이,방향)도 중요하다
- ST-GCN의 partition strategies와 같이 고정된 adjacency matrix를 쓰는 것이 아닌 학습 가능한 adjacency matrix를 추가하자 (label은 동일하게 사용) => 전체 샘플에 대한 것과 특정 샘플(포즈)에 대한 것을 구분하여 계산
- 구조상 Spatial 영역에서 Graph Convolution을 한뒤 Temporal 축으로 Convolution 진행
- Joint Position 외에도 Bone Vector 정보를 활용한 모델을 같이 사용해서 성능 향상

제안 기법 - Adaptive Graph Convolution(AGC)

- ST-GCN은 식 (2)와 같이 GC을 구현함
  => 본 논문은 식 (3)으로 AGC를 구현함 (그림 2)
  - A : ST-GCN에서 사용한 Spatial Configuration 기법으로 물리적인 연결을 표현
  - B : AGC block 별로 학습 가능한 adjacency matrix [Kernel size, Joint num, Joint num]가 존재
    => 식(2)에서의 Mask M을 대채
    (Mask도 Attention으로 사용하였으나 adjacency matrix값이 0이면 의미가 없어서 바꿈)
    학습 시작시 0으로 초기화
  - C : input feature를 1x1 convolution layer에 태운뒤 dot product, softmax로 Attention Score계산
    (관련 식 4, 5)
    학습 시작시 0으로 초기화
  - Residual Network 구성

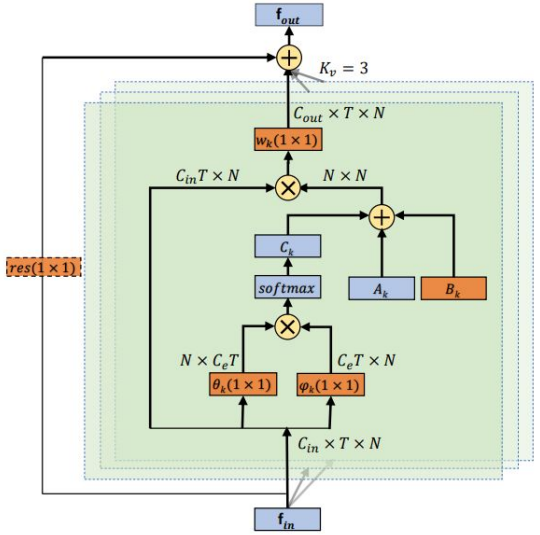

Figure 2. Illustration of the adaptive graph convolutional layer. There are a total of three types of graphs in each layer, i.e., $\mathbf{A}_k$, $\mathbf{B}_k$ and $\mathbf{C}_k$. The orange box indicates that the parameter is learnable. (1×1) denotes the kernel size of convolution. K_v denotes the number of subsets. $\oplus$ denotes the elementwise summation. $\otimes$ denotes the matirx multiplication. The residual box (dotted line) is only needed when C_{in} is not the same as C_{out}.

$$\mathbf{f}_{out} = \sum_k^{K_v} \mathbf{W}_k (\mathbf{f}_{in} \mathbf{A}_k) \odot \mathbf{M}_k \qquad (2)$$

N+N, learnable

$$f(v_i, v_j) = \frac{e^{\theta(v_i)^T \phi(v_j)}}{\sum_{j=1}^{N} e^{\theta(v_i)^T \phi(v_j)}} \qquad (4)$$

$$\mathbf{C}_k = softmax(\mathbf{f_{in}}^T \mathbf{W}_{\theta k}^T \mathbf{W}_{\phi k} \mathbf{f_{in}}) \qquad (5)$$

$$\mathbf{f}_{out} = \sum_k^{K_v=3} \mathbf{W}_k \mathbf{f}_{in} (\mathbf{A}_k + \mathbf{B}_k + \mathbf{C}_k) \qquad (3)$$

N+N

N×T×Cin

learnable

제안 기법

- 그림 3) AGC를 활용하기위해AGC Block의 레이어순서 표현
  Convt 의 경우 시간축을 기준으로 convolution하는 layer
- 그림 4) 저자가 사용한 네트워크 구조를 표현. AGC Block을 겹쳐서 사용
- 그림 5) 기존 논문들은 Joint position 정보를 입력으로 활용했는데 Joint를 이은 본정보도 학습에 필요하다고 생각하여 부모 Joint와 자식 Joint Position간 x,y,z 차이값을 입력으로 사용하는 Bone-Stream을 추가로 사용할 것을 제안한 그림

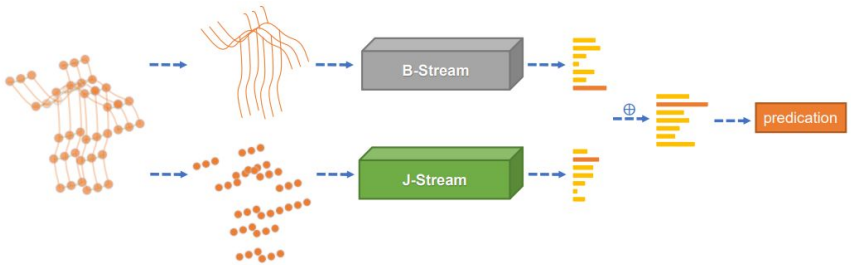

Figure 5. Illustration of the overall architecture of the 2s-AGCN. The scores of two streams are added to obtain the final prediction.

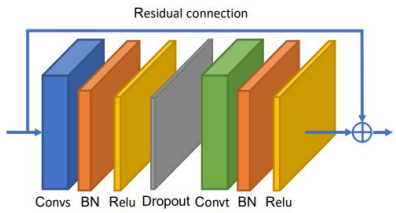

Figure 3. Illustration of the adaptive graph convolutional block. Convs represents the spatial GCN, and Convt represents the temporal GCN, both of which are followed by a BN layer and a ReLU layer. Moreover, a residual connection is added for each block.

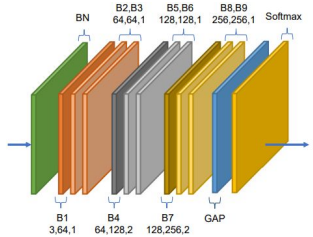

Figure 4. Illustration of the AGCN. There are a total of 9 blocks (B1-B9). The three numbers of each block represent the number of input channels, the number of output channels and the stride, respectively. GAP represents the global average pooling layer.

실험 - 시각화

- 그림 6) 사용한 데이터셋(Kinetics/NTU-RGB)의 skeleton 구조
- 그림 7) 기존 Adjacency matrix 와 Learnable Adjacency matrix 비교
- 그림 8) 같은 sample의 25번째 joint(오른손등)에 대한 3(빨강)/5(파랑)/7(초록)번째 AGCB에서의 adjacency matrix 시각화
  - 원이 큰 joint와 질수록 해당 레이어에서 관계가 있음을 표현
- 그림 9) 다른 sample(빨,파,초)의 25번째 joint에 대한 같은 5번째 AGCB에서의 adjacency matrix 시각화
  - 다른 포즈에 맞춰서 attention이 바뀜

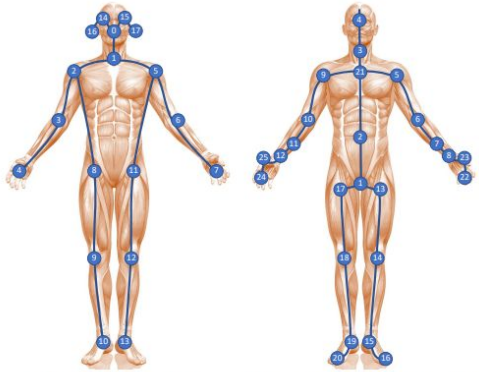

Figure 6. The left sketch shows the joint label of the Kinetics-Skeleton dataset and the right sketch shows the joint label of the NTU-RGBD dataset.

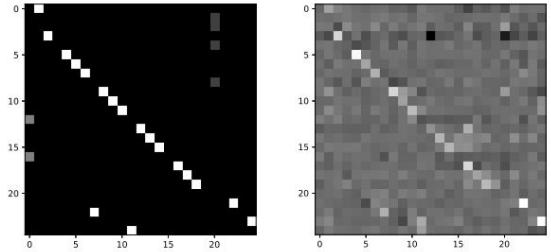

Figure 7. Example of the learned adjacency matrix. The left matrix is the original adjacency matrix for the second subset in the NTU-RGBD dataset. The right matrix is an example of the corresponding adaptive adjacency matrix learned by our model.

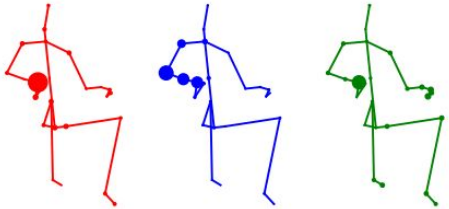

Figure 8. Visualization of the graphs for different layers.

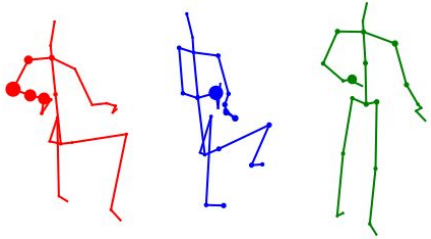

Figure 9. Visualization of the graphs for different samples.

실험 - 모델 성능비교

- 표 1) NTU-RGBD 데이터셋으로 제안 기법 제외했을 때의 성능비교
  - ST-GCN의 learning-rate/data preprocessing을 바꿔서 성능 개선 후 비교실험
  - A,B,C 한개씩 제외했을때 성능차이가 그리 크지 않음
    => 원래 효과가 적거나 서로 충돌이 있거나…
- 표 2) NTU-RGBD 로 joint stream / bone stream / two stream 비교
- 표 3) NTU-RGBD 에서의 다른 모델과 성능비교
- 표 4) Kinetics 데이터 => OpenPose기반의 skeleton 데이터셋에 대한 성능비교

Methods	Accuracy (%)
ST-GCN	92.7
ST-GCN wo/M	91.1
AGCN wo/A	93.4
AGCN wo/B	93.3
AGCN wo/C	93.4
AGCN	93.7

Table 1. Comparisons of the validation accuracy when adding adaptive graph convolutional block with or without A, B and C. wo/X means deleting the X module.

Methods	X-Sub (%)	X-View (%)
Lie Group [31]	50.1	82.8
HBRNN [6]	59.1	64.0
Deep LSTM [27]	60.7	67.3
ST-LSTM [22]	69.2	77.7
STA-LSTM [29]	73.4	81.2
VA-LSTM [33]	79.2	87.7
ARRN-LSTM [19]	80.7	88.8
Ind-RNN [20]	81.8	88.0
Two-Stream 3DCNN [21]	66.8	72.6
TCN [14]	74.3	83.1
Clips+CNN+MTLN [13]	79.6	84.8
Synthesized CNN [23]	80.0	87.2
CNN+Motion+Trans [18]	83.2	89.3
3scale ResNet152 [17]	85.0	92.3
ST-GCN [32]	81.5	88.3
DPRL+GCNN [30]	83.5	89.8
2s-AGCN (ours)	88.5	95.1

Table 3. Comparisons of the validation accuracy with state-of-the-art methods on the NTU-RGBD dataset.

Methods	Top-1 (%)	Top-5 (%)
Feature Enc. [8]	14.9	25.8
Deep LSTM [27]	16.4	35.3
TCN [14]	20.3	40.0
ST-GCN [32]	30.7	52.8
Js-AGCN (ours)	35.1	57.1
Bs-AGCN (ours)	33.3	55.7
2s-AGCN (ours)	36.1	58.7

Table 4. Comparisons of the validation accuracy with state-of-the-art methods on the Kinetics-Skeleton dataset.

Methods	Accuracy (%)
Js-AGCN	93.7
Bs-AGCN	93.2
2s-AGCN	95.1

Table 2. Comparisons of the validation accuracy with different input modalities.