요약

[1] (2018)Skeleton-Based Action Recognition with Spatial Reasoning and Temporal Stack Learning

- 저자가 생각하는 human skeleton sequence의 특징
  - 1) skeleton frame에 풍부한 신체 구조 정보가 포함되도록 각 노드와 인접 노드간에 강한 상관관계가 있음
  - 2) 시간적 연속성은 동일한 joint뿐만 아니라 body structure에도 존재
  - 3) 공간적 영역과 시간적 영역 사이에는 co-occurrence relationship이 있음
- 논문 [1]으로 이 특징들을 담을 수 있는 모델을 만들려고 Graph와 LSTM을 혼합해서 제안했으나 특징 (3)을 충족하기 위해서 본 논문을 제안함
- 주요 모델 구조
  - 입력으로 들어온 현재 프레임의 feature와 이전 프레임 feature와의 차이값을 concat하여 LSTM을 사용하는 Joint Feature Representation - 그림 (2)의 Feature Augmentation
  - 기존 LSTM의 gate에 Graph Convolution 추가 및 Bahdanau Attention을 섞은 Attention Enhanced Graph Convolutional LSTM(AGC-LSTM) 제안
  - AGC-LSTM 모듈의 출력값으로 Attention Score를 무시한 Global Feature와 Attention Score로 weighted sum한 Local Feature로 Classification

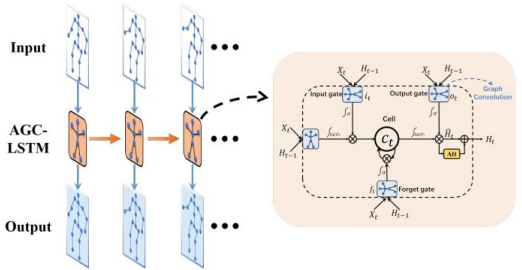


Figure 1. The structure of one AGC-LSTM layer. Different from traditional LSTM, the graph convolutional operator within AGC-LSTM causes the input, hidden state, and cell memory of AGC-LSTM to be graph-structured data.

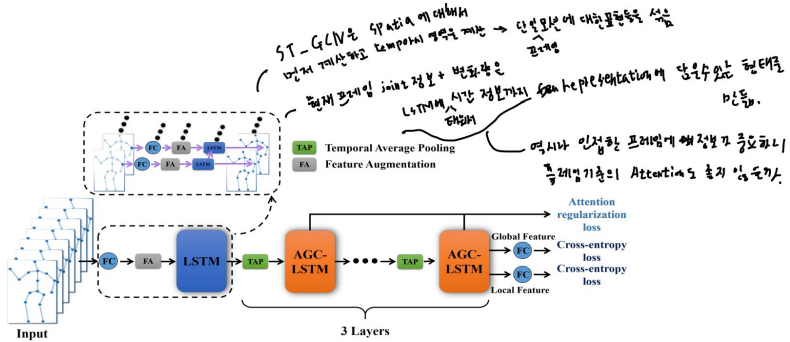
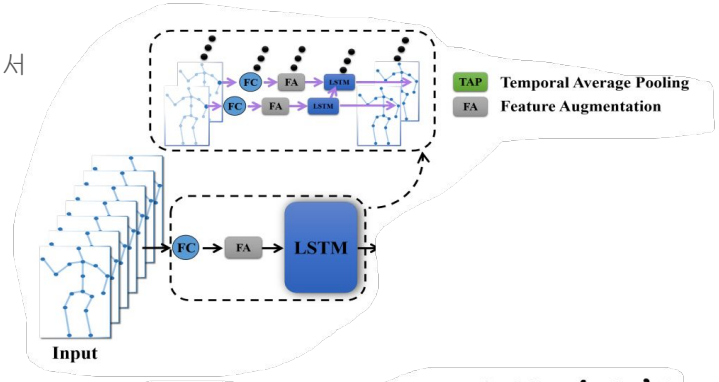


Figure 2. The architecture of the proposed attention enhanced graph convolutional LSTM network (AGC-LSTM). Feature augmentation (FA) computes feature differences with position features and concatenates both position features and feature differences. LSTM is used to dispel scale variance between feature differences and position features. Three AGC-LSTM layers can model discriminative spatial-temporal features. Temporal average pooling is the implementation of average pooling in the temporal domain. We use the global feature of all joints and the local feature of focused joints from the last AGC-LSTM layer to predict the class of human action.

Joint Feature Representation / Temporal Average pooling

- Joint Feature Representation
  - 입력 데이터 [프레임 수, 조인트 수, 3d position] 를 Linear layer와 LSTM layer에 태워서 high-dimensional feature space에 맵핑함 => 식 8)
  - 입력 데이터를 Linear layer(output_dim=256)에 태운 뒤, time t의 feature와 t-1 feature를 element-wise subtraction한 결과값을 time t의 feature와 concat
  - 원래 feature와 이전 프레임과의 차이값은 분산이 다를테니 LSTM에 넣어서 해결
  - LSTM/Linear는 모든 joint에 동일한 weight를 적용함



$$\mathbf{E}_{ti} = f_{lstm}\left(concat\left(\mathbf{P}_{ti}, \mathbf{V}_{ti}\right)\right)$$
$$= f_{lstm}\left(concat\left(\mathbf{P}_{ti}, \left(\mathbf{P}_{ti} - \mathbf{P}_{(t-1)i}\right)\right)\right) \quad (8)$$

$E_t \in R^{N \times D_e}$

256+256=512 Dimension

- Temporal Average Pooling
  - 모든 프레임에 대해서 연산하기에는 많은 자원이 필요하므로 (+ 소폭의 정확도 향상) AGC-LSTM 모듈 전에 frame에 대해서 Average Pooling을 진행
  - 시간축으로 receptive field를 늘려주는 효과가 생김(더 멀리 떨어진 시간선과의 LSTM 가능)
  - 사담) motion search를 하기위한 DTW 기법에서도 전체 프레임이 아닌 일부 프레임만 비교하여 성능향상된 논문이 있음 => “(2020)Keys for Action: An Efficient Keyframe-Based Approach for 3D Action Recognition Using a Deep Neural Network”

AGC-LSTM

- 기존 LSTM의 gate 영역에 graph convolution을 끼워넣음
  => 식 (3)으로 LSTM에 GC를 추가
- LSTM의 Hidden state값에 node에 대한 Attention도 넣어서 spatial attention network를 추가-그림(4)
  - 식 (5)로 어텐션 스코어 계산 - Bahdanau Attention 사용
    - 주요 joint가 여러개일 수 있도록 Attention Score 계산에 Softmax가 아닌 Sigmoid를 사용
- 모델 내의 모든 AGC-LSTM의 끝에 모든 노드(joint) feature의 Aggregation을 Global Feature, Attention score로 Weighted sum한 것을 Local Feature를 계산하고, Action Classification에 사용
  => Linear Layer에 넣음
- 실험에서는 ST-GCN와 동일한 조건에서 정확도 비교를 하기위해 ST-GCN이 사용한 Spatial Configuration을 사용함
  거리 1, 계층에 따라 부모노드/자신노드/자식노드의 label을 다르게하고 label에 따라 convolution 진행

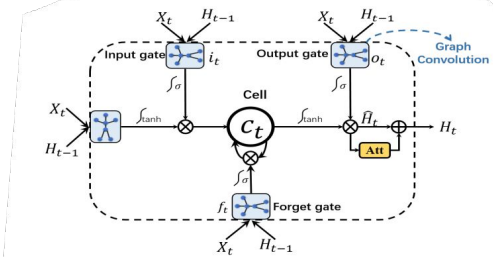


Figure 3. The structures of AGC-LSTM unit. Compared with LSTM, the inner operator of AGC-LSTM is graph convolutional calculation. To highlight more discriminative information, the attention mechanism is employed to enhance the features of key nodes.

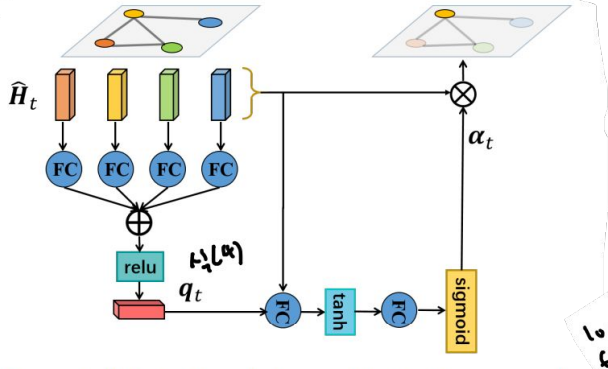


Figure 4. Illustration of the spatial attention network.

$$\mathbf{Y}_{out}(v_{ti}) = \sum_{v_{tj} \in \mathcal{N}(v_{ti})} \frac{1}{Z_{ti}(v_{tj})} \mathbf{X}(v_{tj}) \mathbf{W}(\ell(v_{tj})) \quad (1)$$

$$\alpha_t = Sigmoid\left(\mathbf{U}_s tanh\left(\mathbf{W}_h \widehat{\mathbf{H}}_t + \mathbf{W}_q \mathbf{q}_t + \mathbf{b}_s\right) + \mathbf{b}_u\right) \quad (5)$$

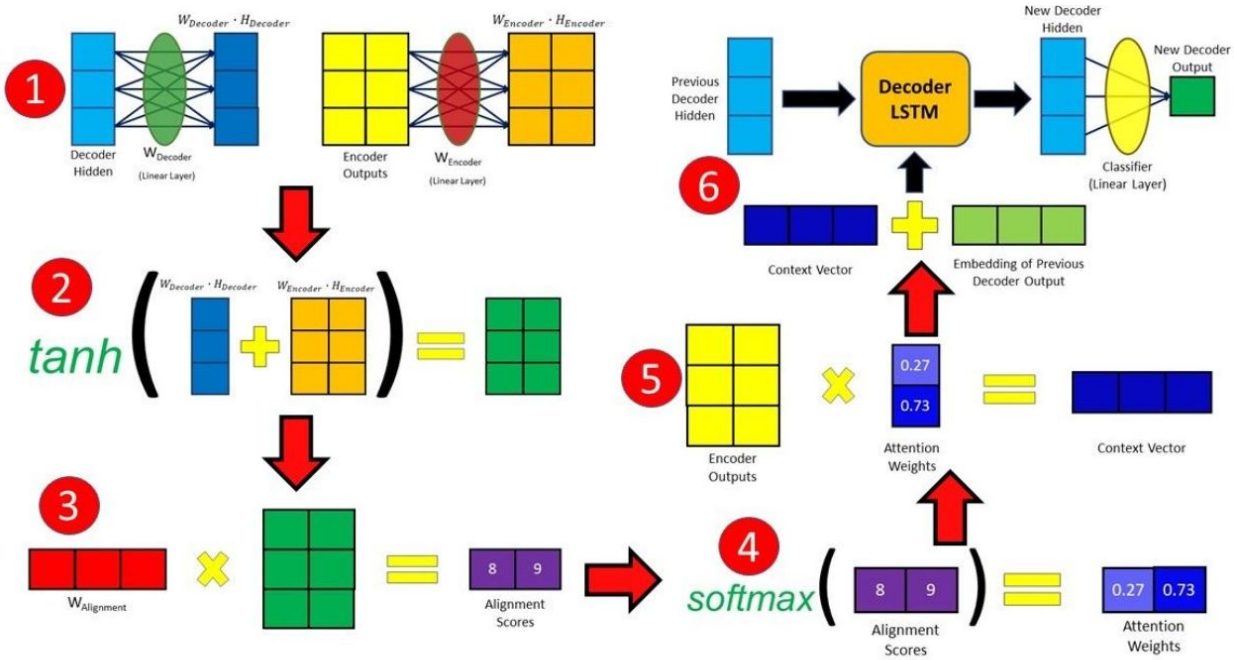
$$\mathbf{F}_t^g = \sum_{i=1}^{N} \mathbf{H}_{ti} \quad (6)$$

$$\mathbf{F}_t^l = \sum_{i=1}^{N} \alpha_{ti} \cdot \widehat{\mathbf{H}}_{ti} \quad (7)$$

$$\begin{aligned}
\mathbf{i}_t &= \sigma(\mathbf{W}_{xi} *_{\mathcal{G}} \mathbf{X}_t + \mathbf{W}_{hi} *_{\mathcal{G}} \mathbf{H}_{t-1} + \mathbf{b}_i) \\
\mathbf{f}_t &= \sigma(\mathbf{W}_{xf} *_{\mathcal{G}} \mathbf{X}_t + \mathbf{W}_{hf} *_{\mathcal{G}} \mathbf{H}_{t-1} + \mathbf{b}_f) \\
\mathbf{o}_t &= \sigma(\mathbf{W}_{xo} *_{\mathcal{G}} \mathbf{X}_t + \mathbf{W}_{ho} *_{\mathcal{G}} \mathbf{H}_{t-1} + \mathbf{b}_o) \\
\mathbf{u}_t &= tanh(\mathbf{W}_{xc} *_{\mathcal{G}} \mathbf{X}_t + \mathbf{W}_{hc} *_{\mathcal{G}} \mathbf{H}_{t-1} + \mathbf{b}_c) \\
\mathbf{C}_t &= \mathbf{f}_t \odot \mathbf{C}_{t-1} + \mathbf{i}_t \odot \mathbf{u}_t \\
\widehat{\mathbf{H}}_t &= \mathbf{o}_t \odot tanh(\mathbf{C}_t) \\
\mathbf{H}_t &= f_{att}\left(\widehat{\mathbf{H}}_t\right) + \widehat{\mathbf{H}}_t
\end{aligned} \quad (3)$$

(참고)Bahdanau Attention

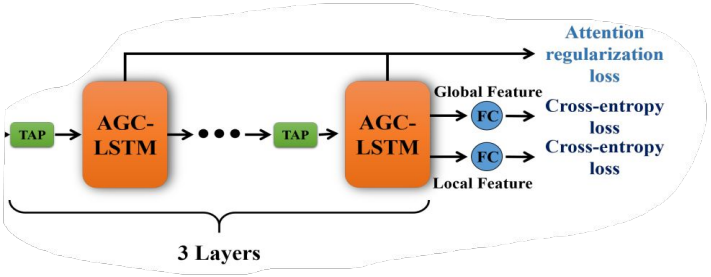
- NLP 기준 다음 단어를 예측할 때 현재 Decoder Hidden state를 Query, Encoder Output Key로 하여 Attention을 진행
- Attention 결과로 만들어진 Context Vector(6번)과 Decoder Output을 더해서 다음 시간 t+1에서의 LSTM 입력으로 사용



Flow of calculating Attention weights in Bahdanau Attention

Loss Function

- 학습에 사용한 Loss Function은 식 (10)과 같음
  - first term) 각 AGC-LSTM의 Global Feature로 action 예측에 대한 Cross-entropy loss
  - second term) 각 AGC-LSTM의 Local Feature로 action 예측에 대한 Cross-entropy loss
  - third term) Attention score가 낮은 joint에 대해서 Loss를 높혀서 주요 joint가 되도록하는 Attention regularization loss
  - fourth term) time별 모든 joint의 Attention Score 합을 loss로 사용해서 attention score를 낮추는 Attention regularization loss



$$\mathcal{L} = -\sum_{t=1}^{T_3}\sum_{i=1}^{C} y_i log\hat{y}_{ti}^{g} - \sum_{t=1}^{T_3}\sum_{i=1}^{C} y_i log\hat{y}_{ti}^{l} \quad (10)$$

$$+\lambda \sum_{j=1}^{3}\sum_{n=1}^{N}\left(1 - \frac{\sum_{t=1}^{T_j}\alpha_{tnj}}{T_j}\right)^2 + \beta \sum_{j=1}^{3}\frac{1}{T_j}\sum_{t=1}^{T_j}\left(\sum_{n=1}^{N}\alpha_{tnj}\right)^2$$

AGC-LSTM 모듈 맨 끝단에 $\hat{y}_t^g$, $\hat{y}_t^l$를 뽑아서 Loss로 사용.

시간x joint x 모듈수

Decay

Decay

모듈별 시간x joint 스코어를 시간별 평균 → joint별 스코어

어텐션 스코어가 낮은 joint에 좀더 어텐션 스코어를 높이도록 학습.

시간x joint

(시간별 스코어 종합)^2

을 시간에 대한 평균을 내고 모듈별로 종합. → Attention score가 너무 높게 되는 것을 방지

실험

- 타 기법들과 성능 비교
  - 같은 데이터셋으로 다른 논문의 실험을 했으면 논문의 결과 값을 그대로 채용
  - NTU RGB+D
    - 표 (1) - SOTA 모델들과 Cross-View, Cross-Subject 정확도 비교
    - 표 (3) - Temporal average pooling 유무(TH)에 따른 정확도 변화
      + Joint Feature Layer 이후 의미있는 관절(팔끼리,다리끼리)끼리 묶은 Graph를 사용하는 Part-based approach 도 실험
  - Northwestern-UCLA
    - 표 (2) - SOTA 모델들과 정확도 비교
    - 표 (4) - 표(3)과 동일한 실험

Methods	Year	CV	CS
HBRNN-L [5]	2015	64.0	59.1
Part-aware LSTM [21]	2016	70.3	62.9
Trust Gate ST-LSTM [16]	2016	77.7	69.2
Two-stream RNN [30]	2017	79.5	71.3
STA-LSTM [25]	2017	81.2	73.4
Ensemble TS-LSTM [13]	2017	81.3	74.6
Visualization CNN [17]	2017	82.6	76.0
VA-LSTM [41]	2017	87.6	79.4
ST-GCN [39]	2018	88.3	81.5
SR-TSL [22]	2018	92.4	84.8
HCN [14]	2018	91.1	86.5
PB-GCN [26]	2018	93.2	87.5
AGC-LSTM (Joint)	-	93.5	87.5
AGC-LSTM (Part)	-	93.8	87.5
AGC-LSTM (Joint&Part)	-	**95.0**	**89.2**

Table 1. Comparison with the state-of-the-art methods on the NTU RGB+D dataset for Cross-View (CS) and Cross-Subject (CV) evaluation in accuracy.

Methods	Year	Accuracy (%)
Lie group [28]	2014	74.2
Actionlet ensemble [32]	2014	76.0
HBRNN-L [5]	2015	78.5
Visualization CNN [17]	2017	86.1
Ensemble TS-LSTM [13]	2017	89.2
AGC-LSTM (Joint)	-	92.2
AGC-LSTM (Part)	-	90.1
AGC-LSTM (Joint&Part)	-	**93.3**

Table 2. Comparison with the state-of-the-art methods on the Northwestern-UCLA dataset in accuracy.

Methods		Accuracy (%)
Joint	LSTM	70.0
	GC-LSTM	87.5
	LSTM+TH	78.5
	GC-LSTM+TH	89.4
	AGC-LSTM+TH (AGC-LSTM)	92.2
Part	AGC-LSTM+TH (AGC-LSTM)	90.1
AGC-LSTM (Joint&Part)		**93.3**

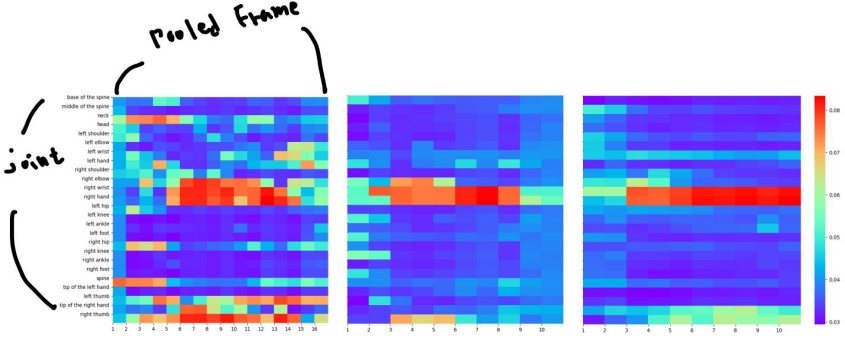
Table 4. The comparison results between several baselines and our AGC-LSTM on the Northwestern-UCLA dataset.

Methods		CV	CS
Joint	LSTM	89.4	80.3
	GC-LSTM	92.4	85.6
	LSTM+TH	90.4	81.4
	GC-LSTM+TH	92.9	86.3
	AGC-LSTM+TH (AGC-LSTM)	93.5	87.5
Part	AGC-LSTM+TH (AGC-LSTM)	93.8	87.5
AGC-LSTM (Joint&Part)		**95.0**	**89.2**

Table 3. The comparison results between several baselines and our AGC-LSTM on the NTU RGB+D dataset.

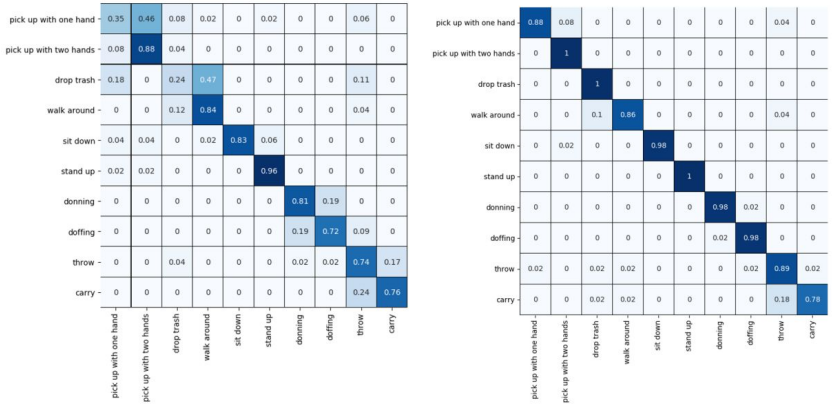
실험 (계속)

- 그림 (6) - “악수” 모션에 대한 AGC-LSTM layer별 Attention Score를 시각화
- 그림 (7) - LSTM과 AGC-LSTM의 confusion matrix 비교
- 그림 (8) - NTU RGB+D 데이터 셋에서 정확도 80%이하의 label에 대한 시각화
  - 읽기/쓰기, 핸드폰/테블릿 사용하기, 키보드 타이핑, 등
    비슷한 움직임을 하는 class에 대해서 잘 틀리는 것을 보임



(a) (b) (c)

Figure 6. Visualizations of the attention weights of three AGC-LSTM layers on one actor of the action “handshaking“. Vertical axis denotes the joints. Horizontal axis denotes the frames. (a), (b), (c) are the attention results of the first, second and third AGC-LSTM layer, respectively.



(a) LSTM (b) AGC-LSTM

Figure 7. Confusion matrix comparison on the Northwestern-UCLA dataset. (a) LSTM. (b) AGC-LSTM.

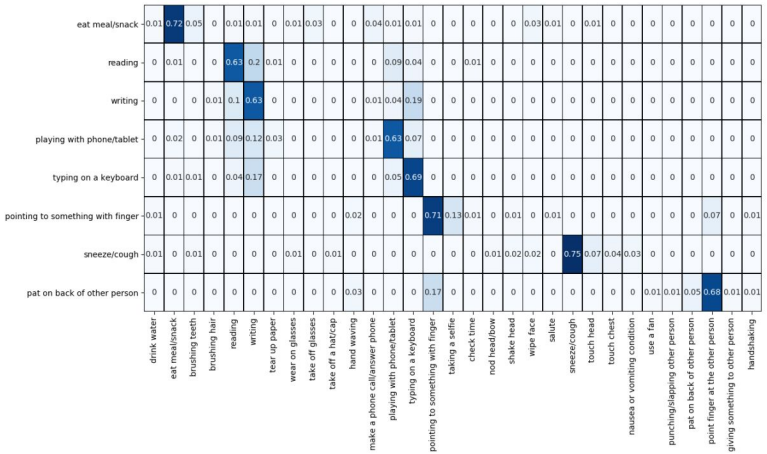


Figure 8. Confusion matrix comparison on the NTU dataset. It shows the part of confusion matrix comparison of the actions (“eat meal/snack”, “reading”, “writing”, “playing with phone/tablet”, “typing on a keyboard”, “pointing to something with finger”, “sneeze/cough”, “pat on back of other person”) with accuracies less than 80% on NTU dataset.