Lip Reading
in the wild

Let's start the transformation Video to Text

keicoon(keicoon15@gmail.com)

Intro

NLP(natural language Processing)

STT(speach to text)

STT 성능이 너무 구려.. 좋은 방법이 없을까?

lip image만 학습해보자!

응. 잘 안돼~

그럼 둘 다 써보자! (multi-modal)

Why Lip Reading ?

- 수 많은 STT solution들은 존재함
  - xxx stt(google, watson)
- 음성(audio) 데이터만으로는 인식에 한계가 분명함
  - 알파벳(m, n) 발음의 모호성
  - 노이즈가 있거나 음질이 안 좋은 환경
- 그래서 입술(image) 데이터를 추가해보자!

- video(image & audio)데이터를 기반으로 subtitle(text)를 생성하는 것을 목표로 함

- 학습 데이터(en)는 BBC News 영상으로 사용함

- 학습 데이터(ko)는 YTN News 영상으로 사용함

- 첫번째 과제.. 그래서 데이터는 누가 만들어주는데?

Huge Video Data

- 많은 비디오 데이터를 수집하고 정제하는 과정이 필요함
- 문장(sentence)를 한번에 학습하기는 어렵기 때문에 단어(word)단위로 학습이 필요함
- 그래서 단어 단위의 영상 편집이 필요함
- audio와 lip image와 synchronize가 필요함
- 잘 학습되는 단어 단위로 데이터 필터링 작업이 필요함

- Collecting YouTube Video
- Split video by face validated
- Split audio by silence detection
- Generate subtitle using STT and cross-check
- Extract lip image and audio(mfcc)
- (extra) translate korean to asm

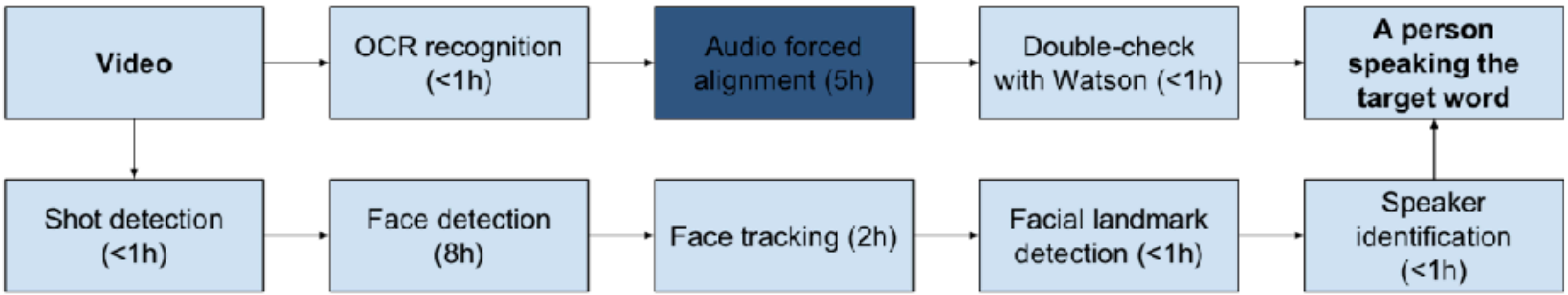


Fig. 2. Pipeline to generate the text and visually aligned dataset. Timings are for a one-hour video.

Model Architecture

– Generate character-sequence

– Models

$$P(\mathbf{y}|\mathbf{x}^v, \mathbf{x}^a) = \prod_i P(y_i|\mathbf{x}^v, \mathbf{x}^a, y_{<i})$$

- Watch Model(encoder)
- Listen Model(encoder)
- Spell Model(decoder)
  - **Dual attention mechanism**

$$\begin{aligned} s^v, \mathbf{o}^v &= \texttt{Watch}(\mathbf{x}^v) \\ s^a, \mathbf{o}^a &= \texttt{Listen}(\mathbf{x}^a) \\ P(\mathbf{y}|\mathbf{x}^v, \mathbf{x}^a) &= \texttt{Spell}(s^v, s^a, \mathbf{o}^v, \mathbf{o}^a) \end{aligned}$$

Watch model

$$
\begin{aligned}
f_i^v &= \mathtt{CNN}(x_i^v) \\
h_i^v, o_i^v &= \mathtt{LSTM}(f_i^v, h_{i+1}^v) \\
s^v &= h_1^v
\end{aligned}
$$

– Using lip image in video frame

- *Reverse order*

– CNN model is VGG-M

- Advanced model than VGG

Listen model

$$h_j^a, o_j^a = \texttt{LSTM}(x_j^a, h_{j+1}^a)$$
$$s^a = h_1^a$$

- Using 13-dimentional MFCC features
  - *Reverse order*

Spell model

- LSTM
- Attention-model
- MLP(multi layer perceptron)

$$
\begin{aligned}
h_k^d, o_k^d &= \mathtt{LSTM}(h_{k-1}^d, y_{k-1}, c_{k-1}^v, c_{k-1}^a) \\
c_k^v &= \mathbf{o}^v \cdot \mathtt{Attention}^{\mathtt{v}}(h_k^d, \mathbf{o}^v) \\
c_k^a &= \mathbf{o}^a \cdot \mathtt{Attention}^{\mathtt{a}}(h_k^d, \mathbf{o}^a) \\
P(y_i|\mathbf{x}^v, \mathbf{x}^a, y_{<i}) &= \mathtt{softmax}(\mathtt{MLP}(o_k^d, c_k^v, c_k^a))
\end{aligned}
$$

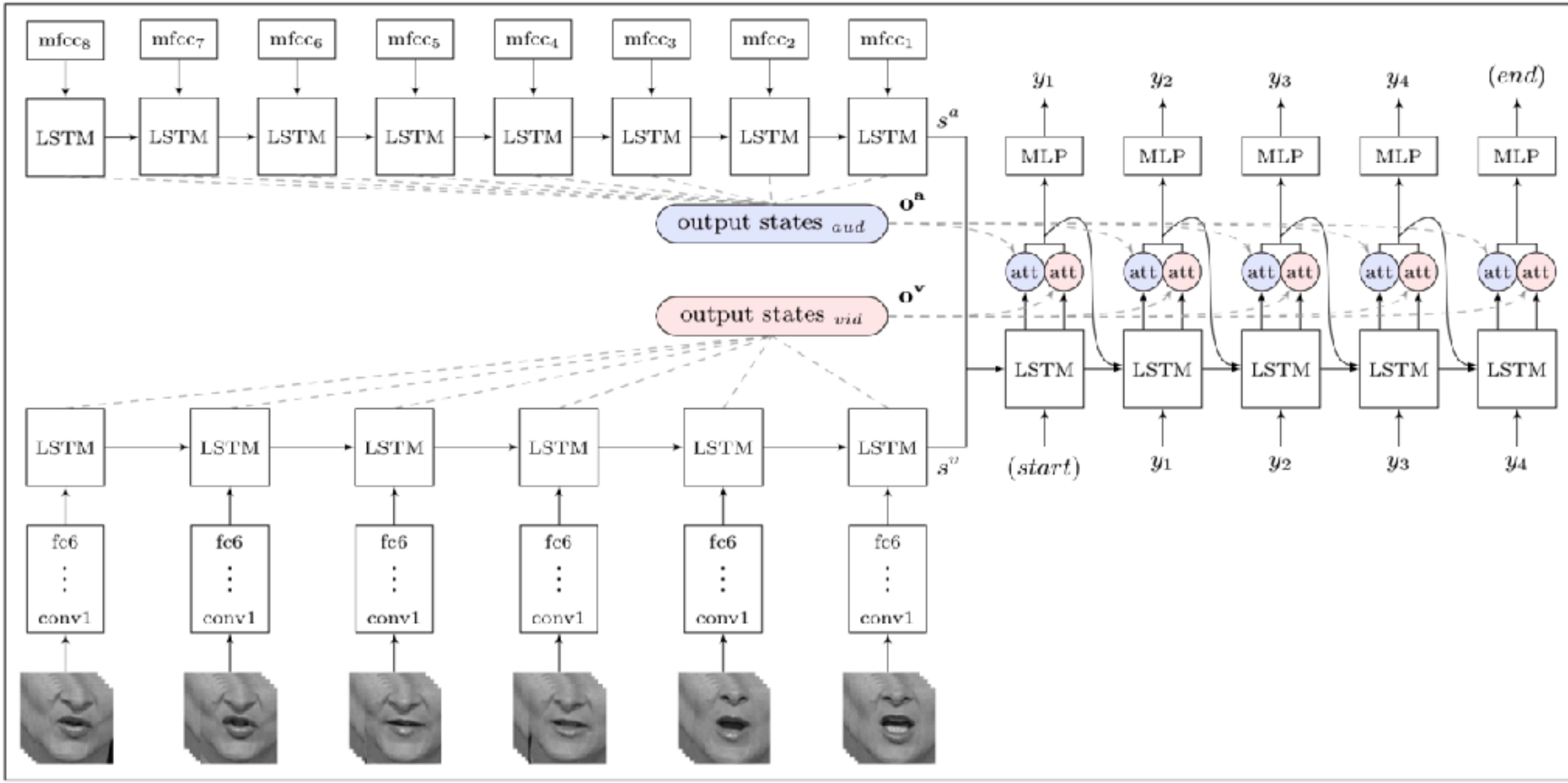
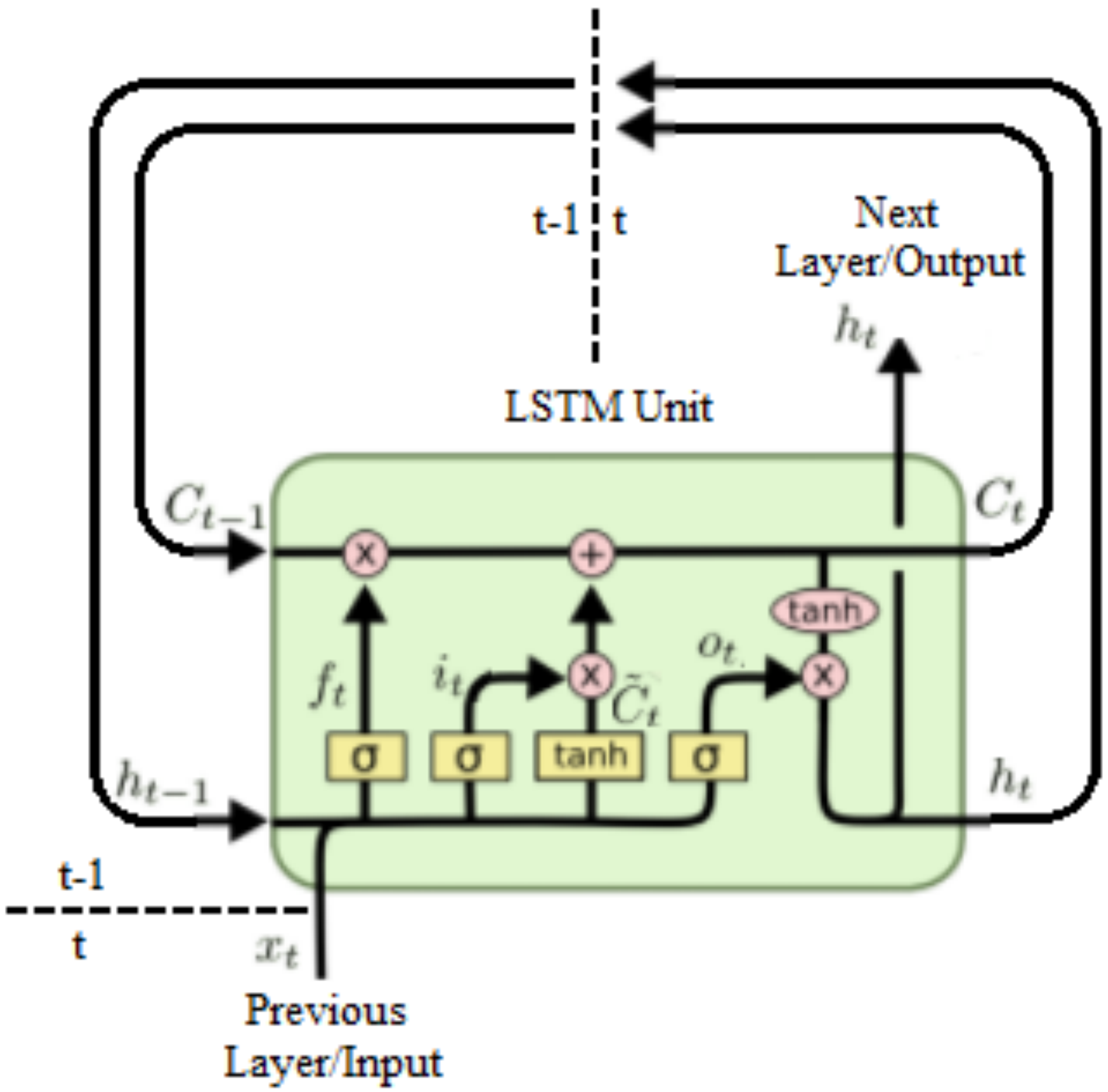
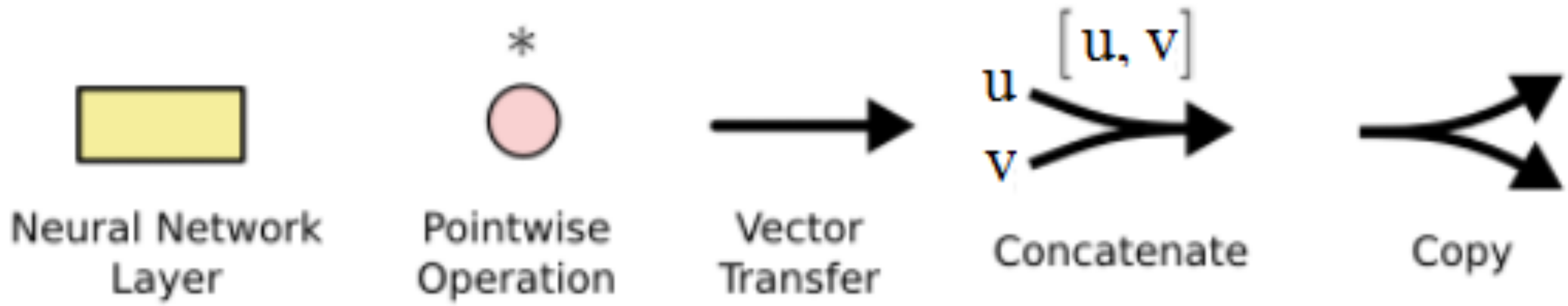


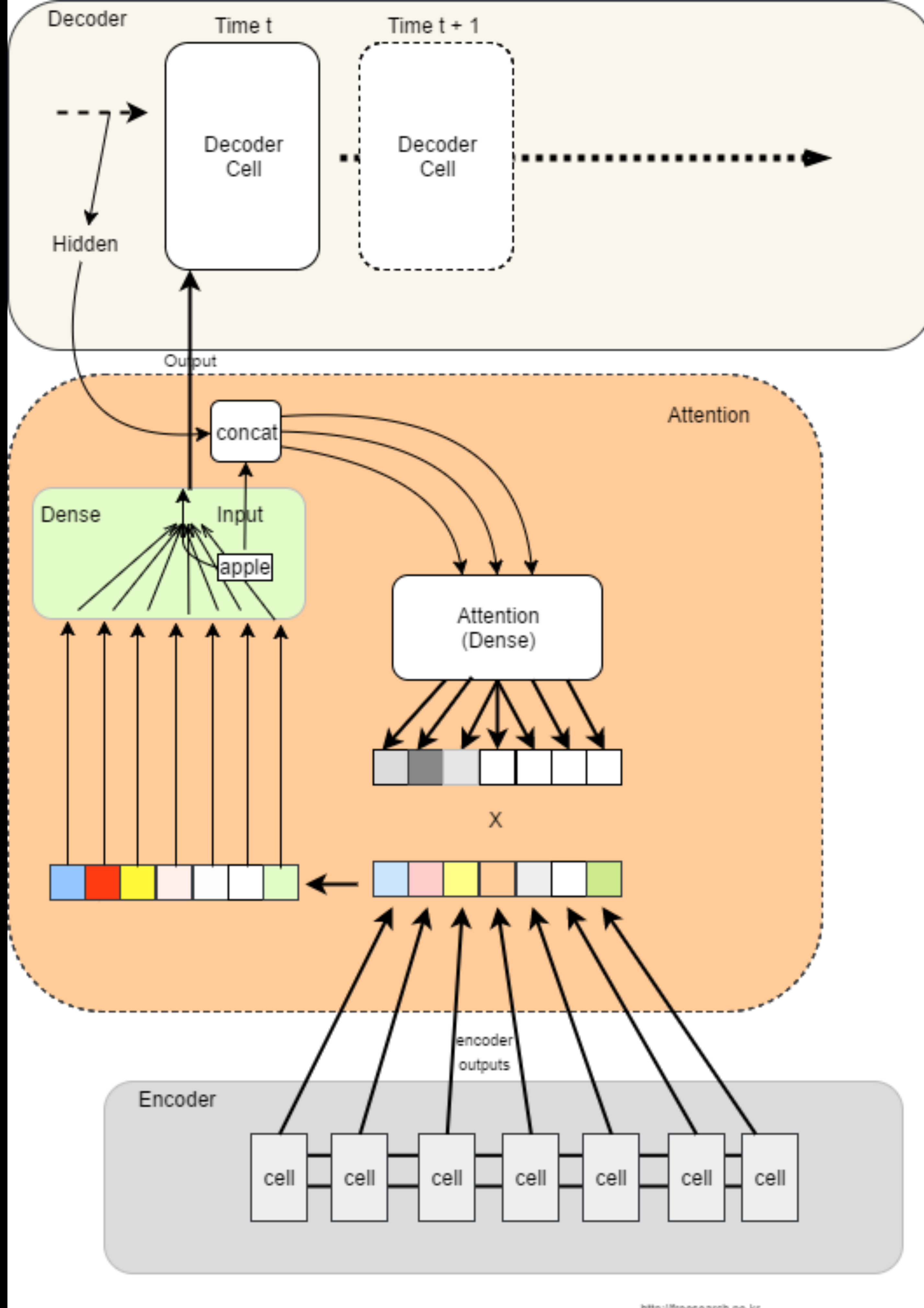
Figure 1. *Watch, Listen, Attend and Spell* architecture. At each time step, the decoder outputs a character y_i, as well as two attention vectors. The attention vectors are used to select the appropriate period of the input visual and audio sequences.

Understanding LSTM Networks



$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \ + \ b_f\right)$$
$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \ + \ b_i\right)$$
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \ + \ b_C)$$
$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$
$$o_t = \sigma\left(W_o \ [h_{t-1}, x_t] \ + \ b_o\right)$$
$$h_t = o_t * \tanh\left(C_t\right)$$




Decoder
Time t
Time t + 1
Decoder Cell
Decoder Cell
Hidden
Output
concat
Attention
Dense
Input
apple
Attention (Dense)
X
encoder outputs
Encoder
cell
cell
cell
cell
cell
cell
cell
http://freesearch.pe.kr

Tranning

- Curriculum Learning
  - Growing word to sentence
- Scheduled sampling
  - Ground-truth in first, and prediction in last
- Multi-modal training
  - Using various mode(audio, lip, both)

Evaluation

- Beam Search
  - More effective decoder
- Protocol
  - CER(character error rate)
  - WER(word error rate)
  - BLEU(bilingual evaluation understudy

Summary

- 기존의 해결법 보다 성능이 조금 더 좋음
- 이미지나 음성이 없는 경우에도 성능이 나옴
- NLP에 관심이 많다면 seq2seq 모델에 관심이 있다면 공부하기 좋은 자료임
- 해결해야하는 점
  - 강세(accent), 발음 속도(speed of speaking), 중얼거림(mumbling)

Appendix

- LipReadingInTheWild(https://www.robots.ox.ac.uk/~vgg/publications/2016/Chung16/chung16.pdf)
- RNN(https://ratsgo.github.io/deep%20learning/2017/04/03/recursive/)
- GetYourFaceVideoData-js(https://github.com/keicoon/GetYourFaceVideoData-js)