**Industry-academia Capstone Design 1**

**Final Report – Korean voice annotation Program**

**Team Name : DD**

**Lee Sin woo, Ko Hun hyeon**

**Dept. of Computer Engineering of Chosun University**

**1. Introduction**

According to the WHO's announcement in 2010, There are 285 million visually impaired people in the world. Of these, 39 million are blind, and 246 million are low vision[1]. Also, according to a 2015 report released by Yonsei University's Institute of Vision Research, The suicide rate of those with low vision is twice that of normal people[2]. Through this, we realized that mental stress caused by low vision is high. We felt that those people needed a service to overcome their physical limitations We're trying to overcome the physical limitations of them.

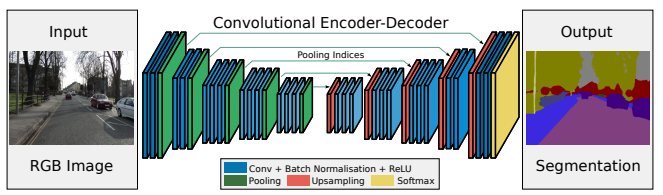
**2. Related Works**

**2.1 Image Segmentation**

Image Segmentation is the segmentation of which class each pixel of the image belongs. The image segmentation can be divided into Smentic Segmentation and Instance Segmentation, where the former classifies objects by pixels, and the latter classifies them by adding object detection.

**2.1.1 SegNet**

SegNet is an image segmentation model that is used for autonomous driving. The model is a combination of encoder and decoder, and the encoder's network uses the Convolution Layer except for the fully connected layer in VGG16. Decoder's network performs Upsampling and Convolution operations, and softmax classifiers are placed on the last layer[1].

****

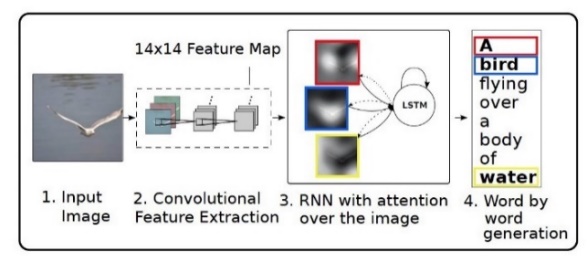
**Fig 1** Architecture of SegNet

**2.2 Image Captioning**

Image captioning is a technique that produces sentences describing images, extracting features of images using encoder and passing the extracted features to decoder to receive word vector sequence as output. The output sequences are matched with word dictionary and eventually printed in human recognizable sentences.

**2.2.1 Show attend and tell**

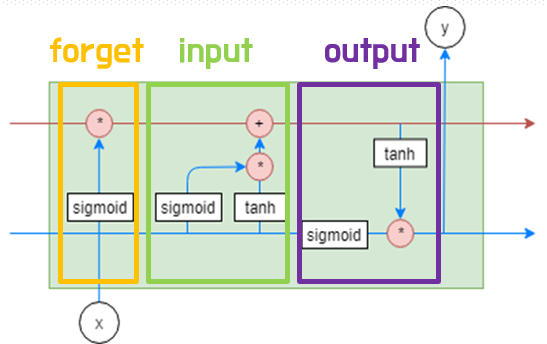
Show attend and tell (Xu et al, 2016) is one of the image captioning model that consists of encoder and decoder and suggests an Attention model that allows model to focus on important parts of the image. The encoder part consists of CNN, and the decoder part consists of RNN(LSTM). The attention model calculates contextual information between the feature map from the encoder and the sentence extracted from ground truth sentence and forwards it to the RNN[4].

****

**Fig 2** Architecture of the Image Captioning Model

**2.2.1.1 LSTM**

In the Show attend and tell paper, LSTM, which is used as a decoder, consists largely of Forget Gate, Input Gate, and Output Gate. Forget gate decides whether to forget the previous cell information. Input gate to determine which value to update to the cell. Output gate to select what to export to the output.

****

**그림 3** Architecture of LSTM

**2.2.1.2 Soft Attention**

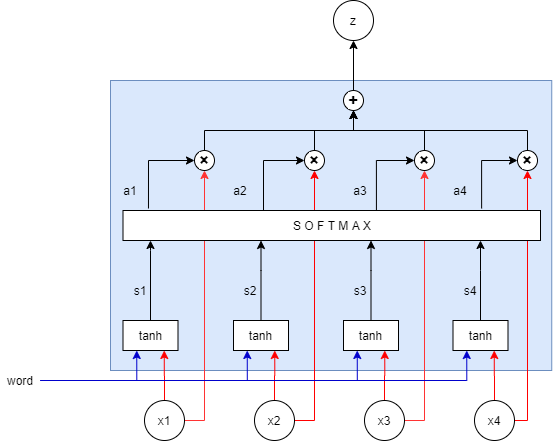
Attention is a method of calculating the attention score with the most similar feature map when outputting sentences from a decoder. Among them, the Soft Attention method is differentiable, so you can calculate the gradient directly. **Fig 4** shows the inputs and outputs of the model. X is an input image, and x1~4 is an encoded image. The attention model calculates the relation between the word and feature map and outputs it as z.

**테이블이(가) 표시된 사진

자동 생성된 설명**

**Fig 4** I/O of the Attention Model

The architecture of the soft-attention model is as shown, and the feature maps x1~x4 in X and the context vector(word) are received as inputs. Then, when passing through the tanh layer, the feature map and the context vector(word) are used to calculate s (Equation 1). At this moment Calculate the corresponding areas only. For example, s1 is identified only by x1 and does not calculate with the others x. Then pass through the softmax layer (Equation 2) and multiply all a and x to get z (Equation 3). This process calculates the relation between the feature map and the sentence of each zone.



**Fig 5** Architecture of the Soft Attention Model

(Equation 1)

(Equation 2)

(Equation 3)

**2.3 Text-to-Speech(TTS)**

TTS is a technology that converts text into speech, and the typical End-to-end TTS are Tacotron2 and Deep Voice2.

**2.3.1 Tacotron2**

Tacotron2 is one of the end-to-end voice synthesis models, which are divided into Embedding, Encoder, Decoder and post-processing parts. First, the character is vectorized through embedding, the feature of the vector is extracted from the encoder, and the mel spectrogram is generated by the decoder. The created mel spectogram is post-processed in a vocoder to produce a wav file format [5].

**스크린샷, 시계이(가) 표시된 사진

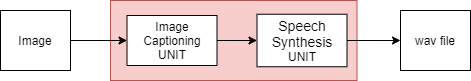
자동 생성된 설명**

**Fig 6** Architecture of Tacotron2

**3. Proposed Method**

**3.1 Architecture overview**

The program was designed by dividing it into the Image Captioning Unit and the Speech Synthesis Unit to create this program that expresses images to voice. The Image Captioning unit receives an image and outputs it as a prediction sentence, and the Speech Synthesis Unit receives a prediction sentence and outputs it as a voice.

****

**Fig 7** Architecture overview

**3.2 Dataset Search**

We’ve been searching the datasets to be used in Image Captioning Unit and Speech Synthesis. And we decided to use MS COCO datasets widely used in Image Captioning and KSS datasets widely used for Korean Speech Synthesis.

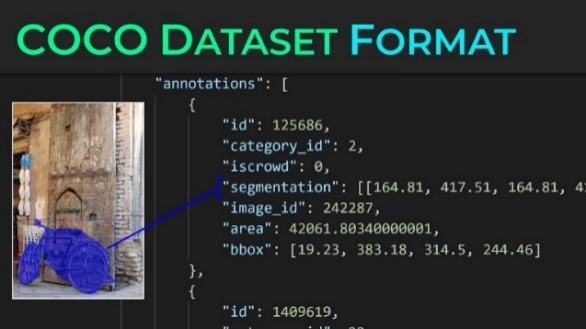
**3.2.1 MS COCO 2017 Dataset**

The MS COCO 2017 dataset is an image captioning database released by Microsoft Research. It consists of a total of 163,957 images, 118,287 in the training set, 5,000 in the validation set, 40,670 in the test set. Except for the test set, the training set and verification set are provided by json files that contain resolution information for each image, five sentences written by a person, and annotation information and caption information for image segmentation.

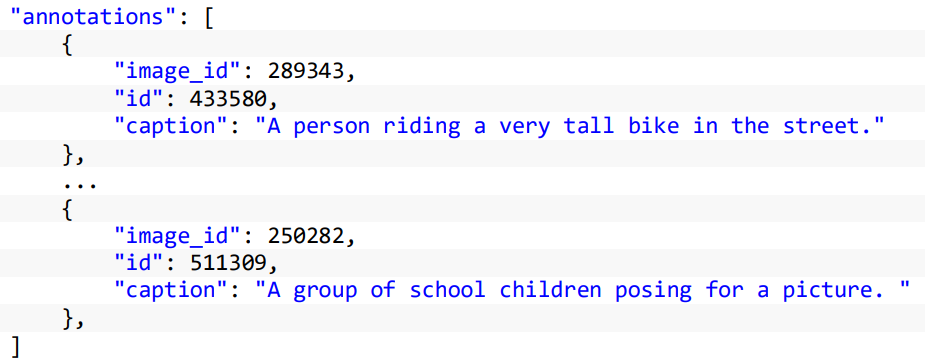
**잔디, 테이블, 여자, 장난감이(가) 표시된 사진

자동 생성된 설명**

**Fig 8** Sample Images of MS COCO Dataset



**Fig 9** Sample of Image Segment Annotation



**Fig 10** Sample of Image Caption Annotation

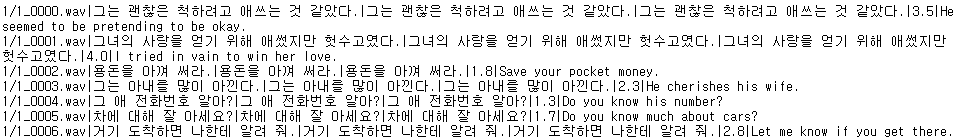
**3.2.2 KSS Dataset**

The KSS data set is a single speaker Korean speech data set being released by Kaggle. It consists of Korean voice actor voice files and dialogues, and a total of 12,853 12-hour voice files are divided into 4 folders. Additionally, Korean scripts for voice files are provided in txt file format.

텍스트, 지도이(가) 표시된 사진

자동 생성된 설명

**Fig 11** Structure of KSS Dataset

****

**Fig 12** KSS Dataset’s Script File

**3.3 Data Preprocessing**

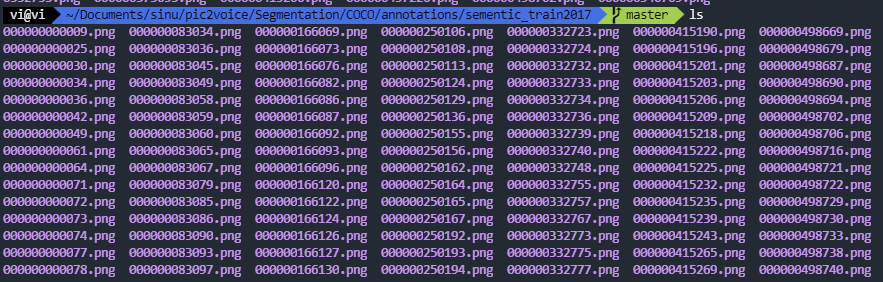
**3.3.1 MS COCO 2017 Dataset Preprocessing**

Since the segmentation mask is required before training the image segmentation model, the segmentation mask is created with MS COCO images and annotation information corresponding to the image. In **Fig 13**, We made a segmentation mask using the image on the left, on the right there is generated mask Image. Each class is distinguished by the brightness (0~90) of the mask image.



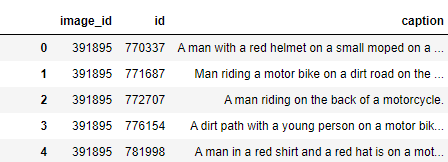
**Fig 13** (**Left**) A Original Image (**Right**) A Generated Mask Image

Segmentation masks for training data and validation data were generated, and a list of some of them is shown in **Fig 14**.



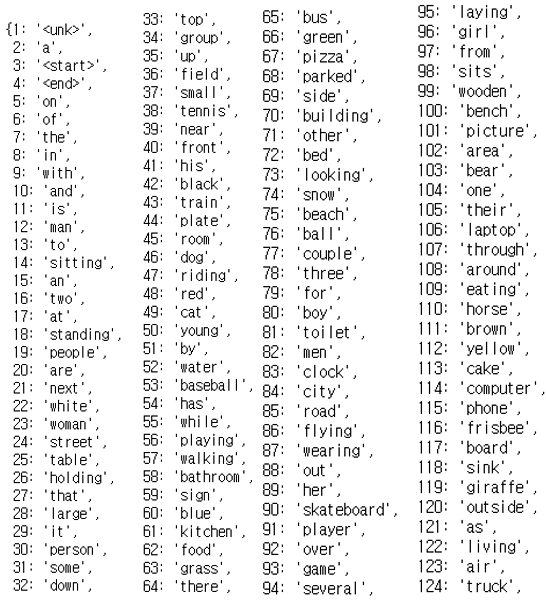
**Fig 14** The list of files that preprocessed MS COCO Dataset

The Image Caption data necessary for learning the Image Captioning model was extracted and saved as a csv file, some of which can be seen in **Fig 15**.



**Fig 15** The list of extracted Image Caption from annotation file

To create a word dictionary for word embedding, a word dictionary was produced with the extracted captions, and some of them are shown in **Fig16**.



**Fig 16** Word dictionary

Word embedding was performed with the produced word dictionary. The process can be seen in **Fig 17**. First, after going through the tokenization process of cutting a sentence into words, it is encoded using the token and word dictionary created. After that, padding was performed to match the maximum length of the sentence.

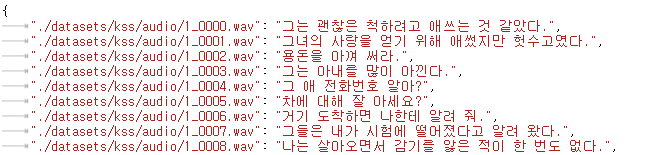
스크린샷이(가) 표시된 사진

자동 생성된 설명

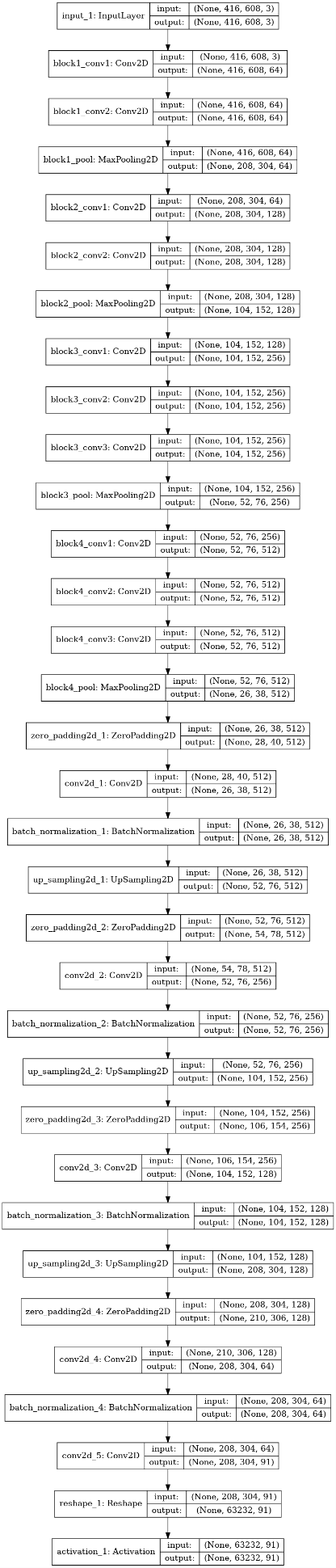
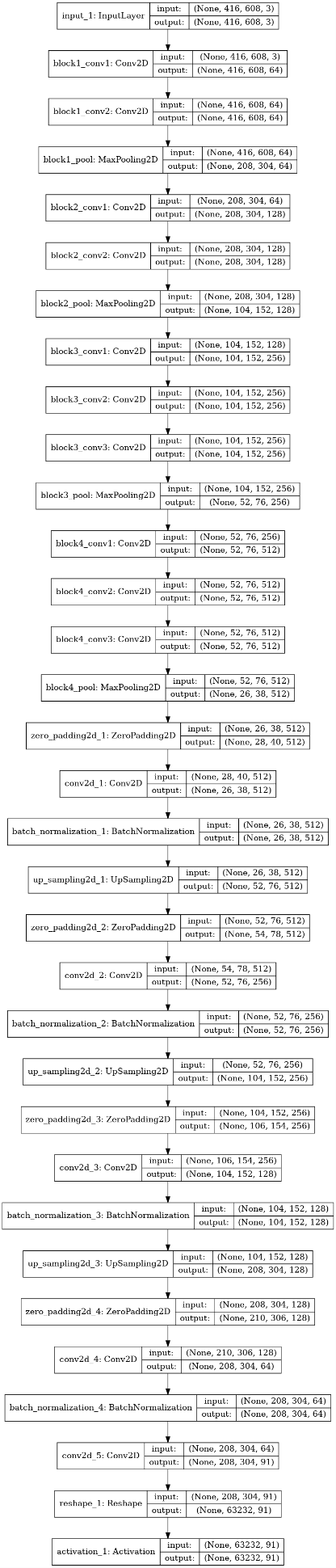
**Fig 17** Word Embedding

**3.3.2 KSS Dataset Preprocessing**

Since the script file is received from Tacotron2 in json format, the script file is modified to the json file format with the macro function of Notepad++.

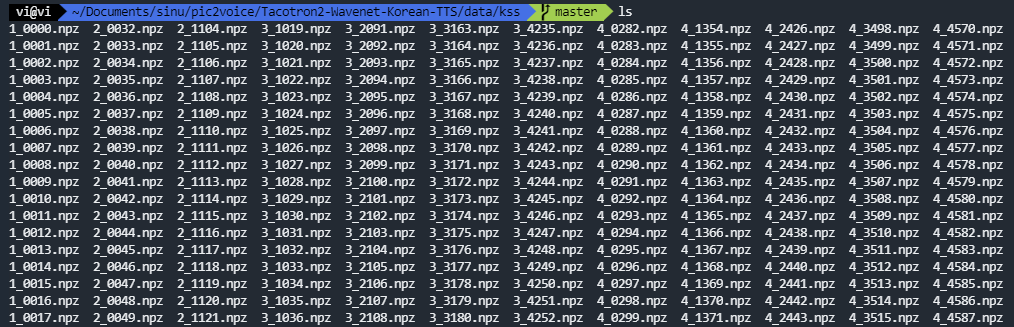


**Fig 18** Convert script file to json format

** **

**Fig 21** Encoder of the Image Segmentation Model **Fig 22** Decoder of the Image Segmentation Model

The rest of the pre-processing except the script file is provided by Tacotron2, so it was used as it is. Preprocessing of KSS data was performed using this. The preprocessing file is preprocessed as npz files by bundling 'audio', 'mel', 'linear', 'time\_steps', 'mel\_frames', 'text', 'tokens', 'loss\_coeff'. Some of the generated npz files list are shown in **Fig 19**.

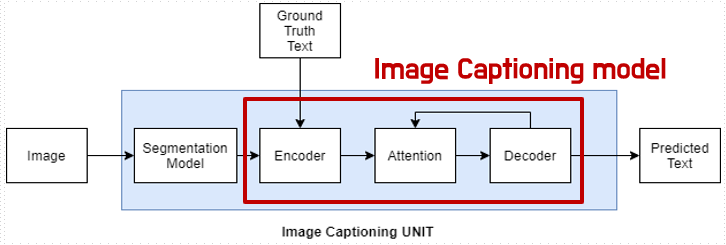


**Fig 19** The list of files that preprocessed the KSS dataset

**3.4 Model description**

**3.4.1 Image Captioning Unit**

The Image Captioning Unit takes the role to create sentences that describe the image. The Image Captioning Unit goes through two steps: model learning and sentence generation. First, learn the model by receiving images and captions. Second, when learning is completed and in service, a prediction sentence is generated with the learned model. In the learning stage, The Image Segmentation Model and the Image Captioning model learn from images and captions.

****

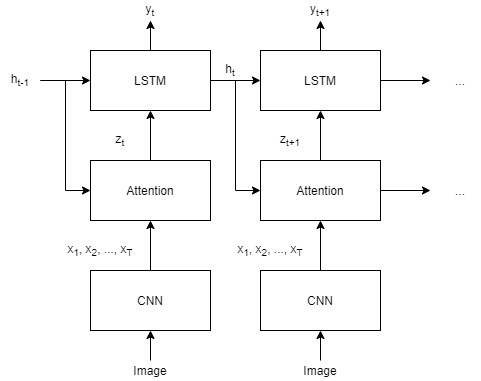
**Fig 20** TheImage Captioning Unit

**3.4.1.1 The Image Segmentation Model**

The Image Segmentation Model classifies image by pixel. The architecture of this model shown in **Fig 21** and **Fig 22**. **Fig 21** shows Encoder of this model and **Fig 22** shows Decoder of the model.

**3.4.1.2 The Image Captioning Model**

The Image Captioning model was used as the encoder, excluding the last fully connected layer of InceptionV3, the attention model used BahdanauAttention, and the cuDNNGRU was used as the decoder. The Image Captioning Model generates an output sentence by calculating the attention score between the feature map extracted from the encoder and the sentence vector of the decoder.

****

**Fig 23** Architecture of the Image Captioning Model

**3.4.2 Tacotron2**

The speech synthesis unit used Tacotron2 speech synthesis model and Griffin Lim vocoder, and plays the role of synthesizing the text output from the Image Captioning Unit into a wav file. The speech synthesis Unit used Tacotron2 because there are already various implementations However, since the vocoder of Tacotron2 uses a vocoder called Wavenet, it takes a long time to learn and generate speech, so a vocoder called Griffin Lim was used in this time. The Speech Synthesis Unit also consists of two steps: model learning and speech synthesis. First, we first receive the voice and the script and then train the model. Second, when the learning is completed, a wav file is created with the trained model.

**스크린샷이(가) 표시된 사진

자동 생성된 설명**

**Fig 24** Architecture of Speech Synthesis Unit

**4. Experiments & Results**

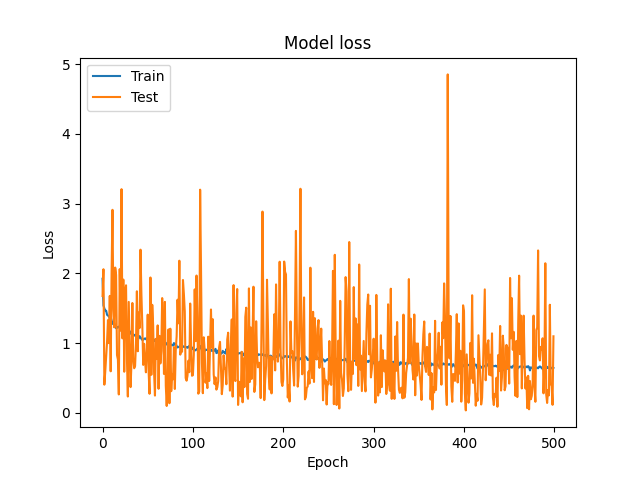
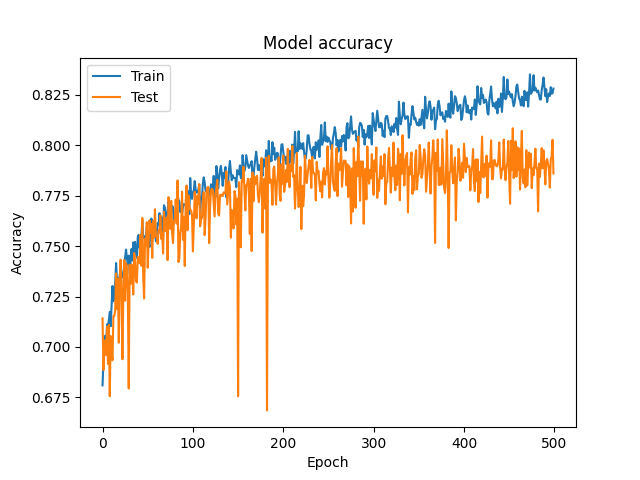
**4.1 Development settings**

The experiment used an Intel i9-9900K processor, GEFORCE RTX 2080 Ti graphics processor, 64GB RAM, Ubuntu 18.04.2 LTS computer. CUDA 10.2 for GPGPU using graphics processor, Docker Community Edition 19.03.8 It was performed after setting the virtual environment with Ubuntu 18.04.3 LTS image. In the virtual environment, python 3.6.9, tensorflow-gpu 1.15.2, and keras 2.3.1 were used for the experiment. For data preprocessing, pandas 1.0.3, numpy 1.18.4, and pycocotools 2.0.0 were used, and opencv-python 4.2.0.34 and imageio 2.8.0 were used for image processing. Nltk 3.2.5 was used for caption data processing, and pypapago 0.1.1.1 was used to translate English captions into Korean. For speech synthesis, librosa 0.6.2, ffmpeg 1.4, and jamo 0.4.1 were used.

**4.2 Experiment**

**4.2.1 The Image Segmentation Model Train**

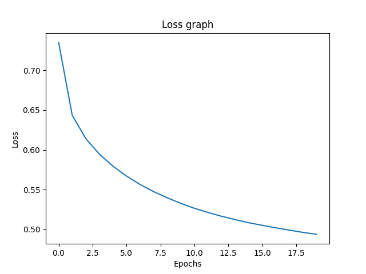
SegNet was trained by 500 epochs using the MS COCO dataset. A graph of model accuracy and loss during training is shown in **Fig 25**.

****

**Fig 25 (Upper)** Accuracy & **(Lower)** Loss graph ofthe Image Segmentation Model

**4.2.2 The Image Captioning Model Train**

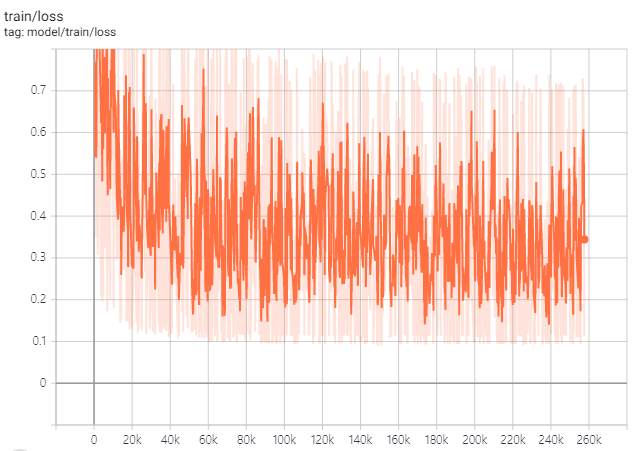
The Image Captioning Model was trained by 20 epochs using the MS COCO dataset. Graphs of model accuracy and loss during training are shown in **Fig 26**.

****

**Fig 26** Loss graph of the Image Captioning Model

**4.2.3 Speech Synthesis Model Train**

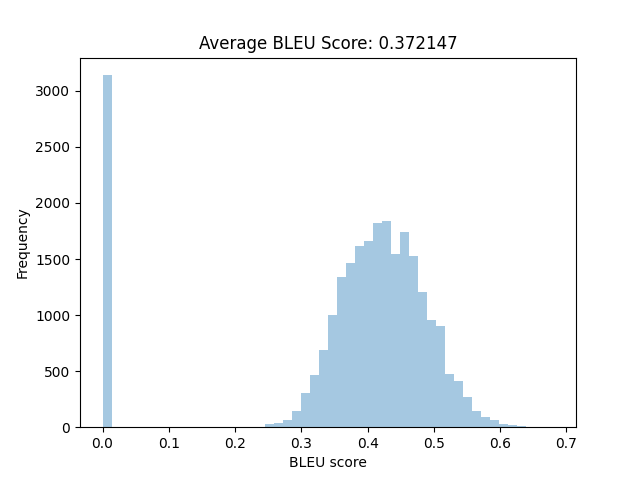
Tacotron2 was trained as much as 257,500epoch, and the model's loss function graph is shown below.



**Fig 27** Loss graph of Tacotron2 model

**4.3 Experimental Result**

To evaluate the proposed method, We used the BLEU score, which is widely used as an indicator of machine translation. Using 25,014 captions of 5000 validation set images, it was measured in comparison with the predicted sentence, and an average BLEU score of 0.372147 was obtained.

****

**Fig 28** BLEU Score Histogram of the Image Captioning Unit

The Image Segmentation Model is shown in **Fig 29** to output the image on the right when the image on the left is entered.

****

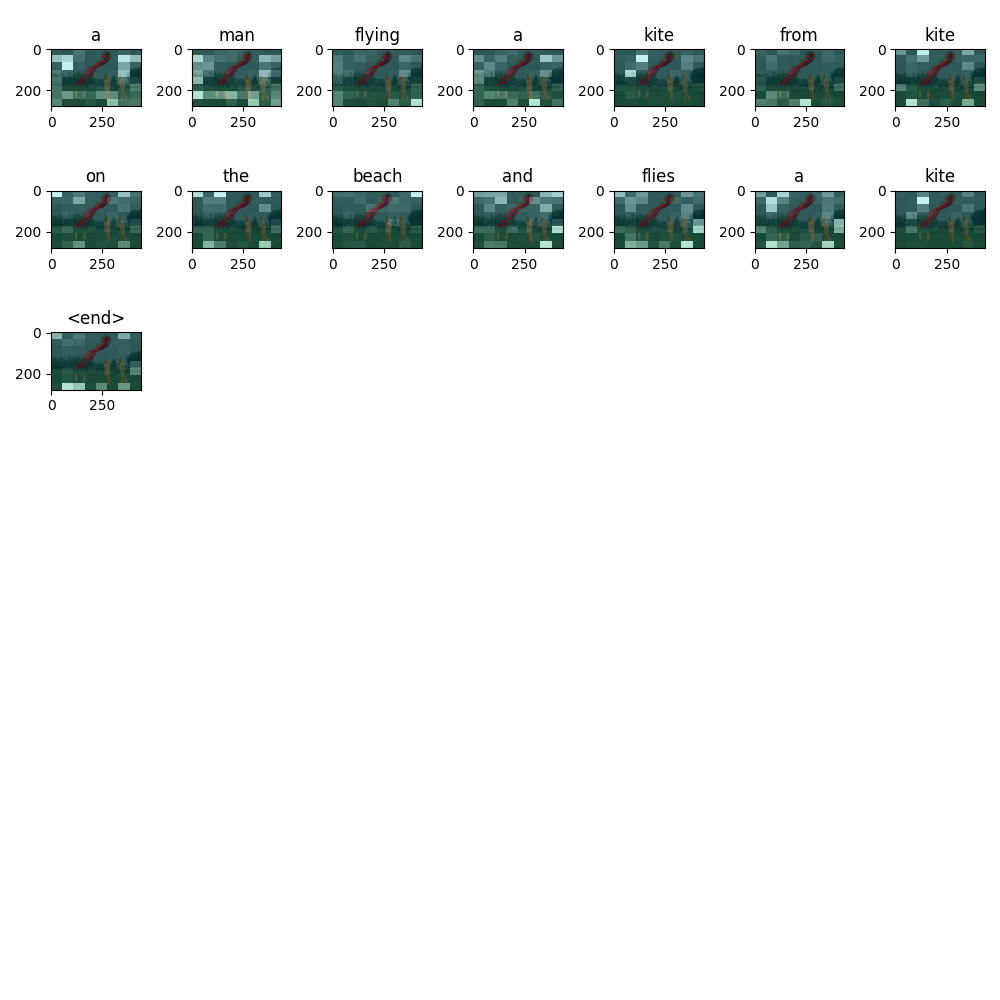
**Fig 29** (**Left**) Input Image (**Right**) Output of the Image Segmentation Model

The input of the Image Captioning Model takes the right image of **Fig 29** (output of the Image Segmentation Model) as input and outputs the predicted sentences in **Fig 30**.

****

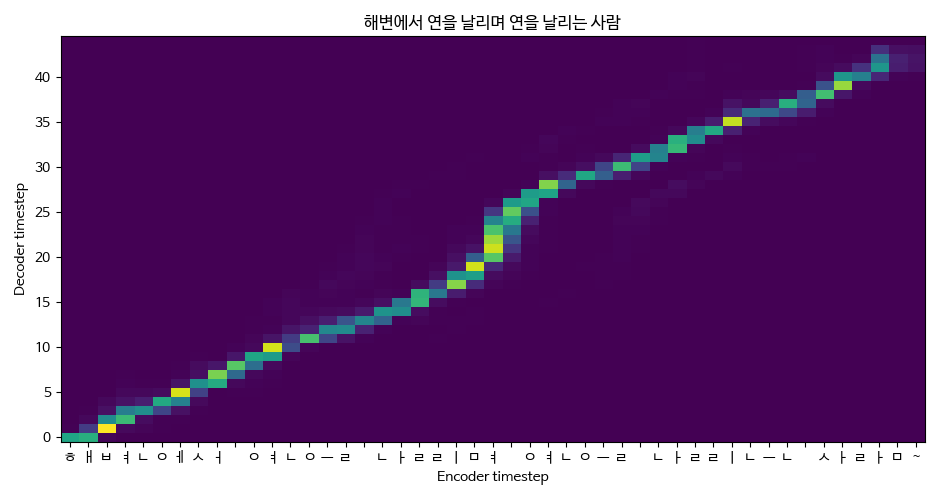
**Fig 30** Output of the Image Captioning Unit

The attention graph of the Image Captioning Model can be found in Fig 31. The graph shows which part of the model focuses on when each word is received as an input, with the intensity of each zone. In other words, the bright part is the place where model focuses on, and the dark part is the place where it is not.

****

**Fig 31** Attention Graph of the Image Captioning Unit

The speech synthesis model receives the caption from the Image Captioning Model as input and outputs the voice file. **Fig 32** shows an attention graph of the voice file. For each timestep, the encoder receives the current state, and outputs a vector indicating the state of the cell at that moment. Decoder generates an audio frame (Mel Spectrogram) using all vectors generated from the encoder. Decoder also determines which vectors on the x-axis are important for creating audio frames at specific moments on each timestep. Here the bright color is the x-axis vector that the decoder focuses on, and the dark part is where it is not.

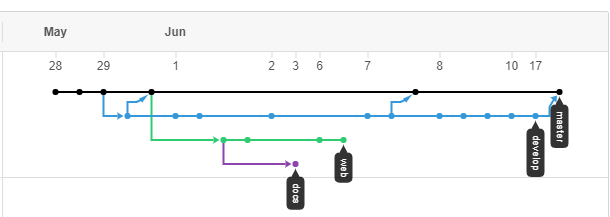
****

**Fig 32** Attention Graph of Speech Synthesis model

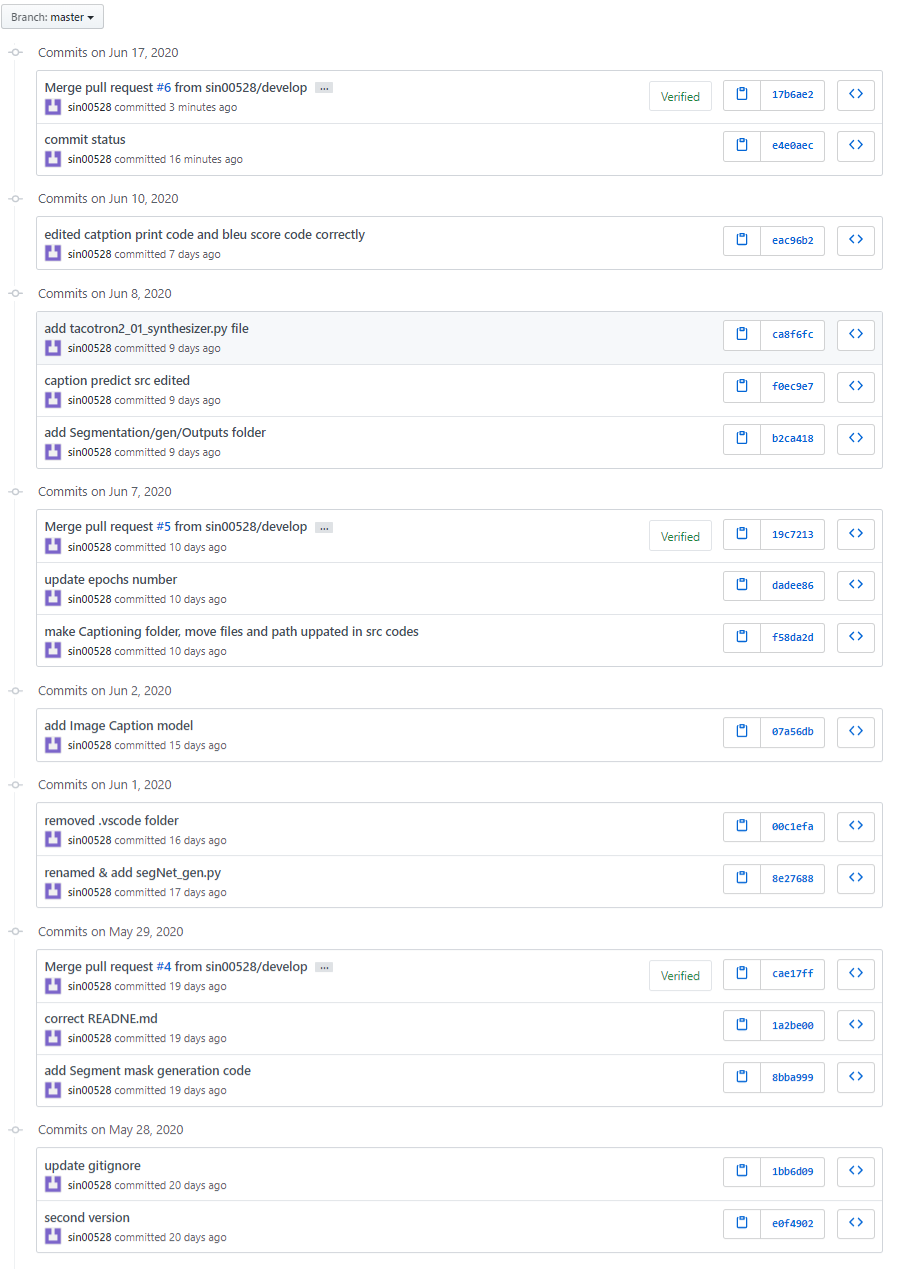
**5. Review & Discussion**

To improve the proposed model, adding morpheme information of captions, and applying various attention techniques could further improve performance. In addition, rather than translating and using English captions, we will consider creating our own Korean captions.

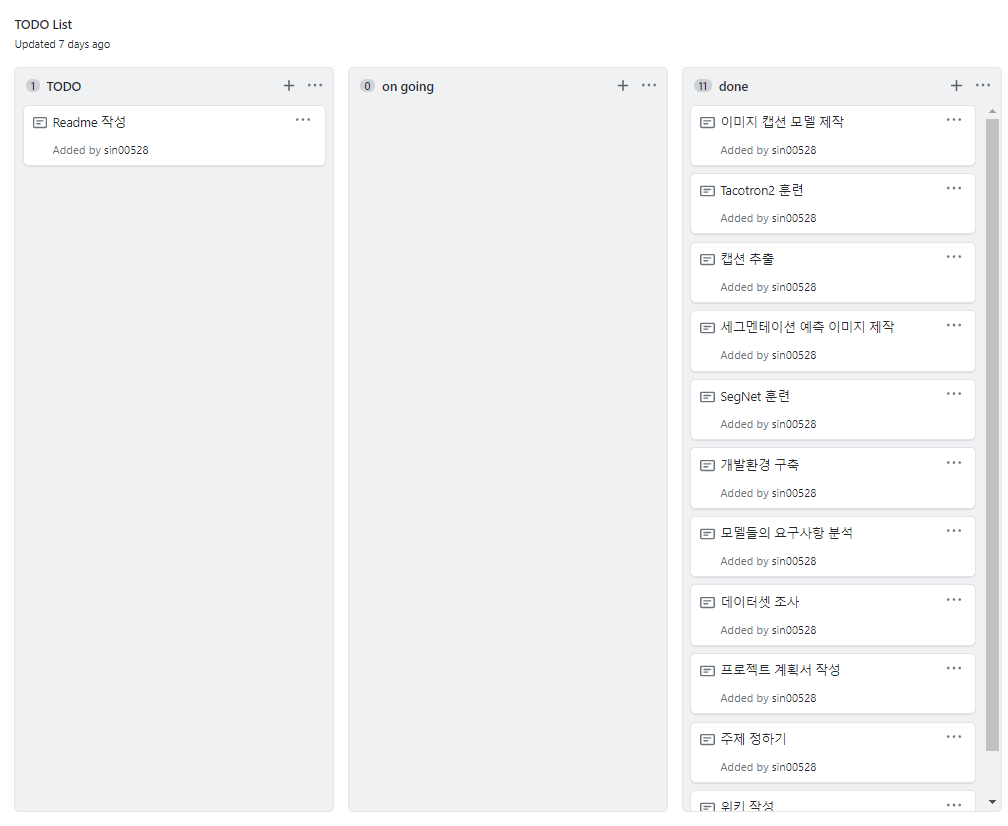
**5. GitHub Logs**

****

**Fig 33** Git Flows

****

**Fig 34** Commit logs of the Master branch



**Fig 35** Scrum board

**6. References**

[1] Pascolini D, Mariotti SP. Global estimates of visual impairment: 2010. Br J Ophthalmol. 2012;96(5):614-618.

[2] Rim TH, Lee CS, Lee SC, Chung B, Kim SS; Epidemiologic Survey Committee of the Korean Ophthalmological Society. Influence of visual acuity on suicidal ideation, suicide attempts and depression in South Korea. Br J Ophthalmol. 2015;99(8):1112-1119.

[3] V. Badrinarayanan, A. Kendall and R. Cipolla, "SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 39, no. 12, 2017, pp. 2481-2495.

[4] Kelvin Xu, Jimmy Lei Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhutdinov, Richard S. Zemel, and Yoshua Bengio. Show, attend and tell: neural image caption generation with visual attention. In Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37 (ICML’15). 2015. pp. 2048–2057.

[5] J. Shen et al., "Natural TTS Synthesis by Conditioning Wavenet on MEL Spectrogram Predictions," 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Calgary, AB, 2018, pp. 4779-4783.