An Multilingual Interactive Picture Book Reader

Submitted 05 2021, in partial fulfilment of
the conditions for the award of the degree
Computer Science with Artificial Intelligence.

14330261
School of Computer Science
University of Nottingham

I hereby declare that this dissertation is all my own work, except as indicated in the text:

Signature _____Yixuan Xu_____

Date ___10__/__05___/__2021___

I hereby declare that I have all necessary rights and consents to publicly distribute this dissertation via the University of Nottingham's e-dissertation archive.*

*Only include this sentence if you do have all necessary rights and consents. For example, if you have including photographs or images from the web or from other papers or documents then you need to obtain explicit consent from the original copyright owner. If in doubt, delete this sentence. See Copyright Information for more details.

**Only include this sentence if there is some reason why your dissertation should not be accessible for some period of time, for example if it contains information which is commercially sensitive or might compromise an Intellectual Property claim. If included, fill in the date from which access should be allowed.

Public access to this dissertation is restricted until: DD/MM/YYY**

*Only include this sentence if you do have all necessary rights and consents. For example, if you have including photographs or images from the web or from other papers or documents then you need to obtain explicit consent from the original copyright owner. If in doubt, delete this sentence. See Copyright Information for more details.

**Only include this sentence if there is some reason why your dissertation should not be accessible for some period of time, for example if it contains information which is commercially sensitive or might compromise an Intellectual Property claim. If included, fill in the date from which access should be allowed.

Abstract

Picture Book Reader (PBR) is the electronic product that replaces parents' roles in reading stories to children. While the significant breakthrough in Computer Vision and Natural Language Processing has been applied as visual and auditory aid for anime and comic book, PBR still relies on pre-record audio in one or two languages which no longer satisfies the young generations. Therefore, the exploration the AI scenarios in the field of children's storybook reading worth consideration. In this project, a Multilingual Interactive Picture Book Reader (MIPBR) is proposed as next generation of PBR. We first implemented the deep learning models to mimic the fairy-tale reading behavior, followed by the interactive user interface in the hope of encouraging children's active intellectual engagement. Lastly we construct a labeling tool that encourages parental involvement while collecting more data for cartoon object detection. The prototype implementation of MIPBR has shown the great potentials of AI in picture book reading.

Acknowledgements

I would like to express my gratitude to my supervisor, Mr. Chao Chen, for his guidance and advice throughout the project.

Contents

1 Introduction

> *"If you want your children to be intelligent, read them fairy tales. If you want them to be more intelligent, read them more fairy tales" —— Albert Einstein[1]*

Fairy-tale reading has survived the constant reformation of the parent-child relationship for centuries. Psychologists and neuroscientists have suggested that fairy-tale reading can provide significant benefit for children's cognitive as well as literary development. Moreover, studies have also shown the positive influence of fairy-tale reading on children's language skills and attention spans[2]. In the modern world, fairy tales are published as storybooks containing attractive and colourful cartoon pictures. Research has shown that high-quality storybooks open a new horizon for pre-school children, sparking their imagination, and more importantly, are equally effective for pre-school children in learning their second language as well as their mother tongue[3].

When it comes to mastering a second language, it is evident that the that the home environment is more important than school in enabling children to become bilingual or multilingual. For example, a family where parents speak different languages, or a family speaks a language that is not the primary language used in their community, are the leading reasons for the children to be bilingual. In this case, storybooks can arouse enough curiosity in children to immerse themselves in the story while the MIPBR can attract children to interact with it in the home environment language. Therefore, it is a product that could satisfy parents in training their children to be multilingual.

Parental involvement is another critical factor in pre-school children's education. Recent research has indicated that newborns are more sensitive to their parent's voices, especially their mother's[4]. Namely, a mother reading stories for her child helps to activate the part of the newborn's brain that is responsible for language learning. Therefore, a mother's voice is said to be the best language learning aid. However, with the accelerating urbanization progress and population growth, fiercer competition in the workplace has resulted in a decreased parental commitment in many child-rearing activities, including fairy-tale reading. A survey has found that, although 60% of parents acknowledged the benefits of reading stories aloud to their children, only 15% out of 1000 parents read stories to their children on a daily basis due to their work schedules and other arrangements[5].

1.1 Motivation

When parents have insufficient time and effort to read stories for their children, they naturally turn to electronic storybook readers on the market that aims to compensate for the parental absence in reading stories. However, most of them have distinct flaws and obsolete functions that no longer satisfy the demands for educating the young generation, especially nurturing modern society talents.

Therefore, the project's motivation is not limited to making up shortcomings of the existing product, but introducing AI to pre-school children's education by defining the next generation of AI-powered multi-lingual storybook reader. By doing so, the young generation is encouraged to keep abreast with technological progress, whereas reading storybooks in other languages can also promote their mutual respect for culture from an early age.

1.2 Aim & Objectives

The aim of this project is to build an interactive reading assistant with instructive functions, enabling pre-school children to develop language skills comprehensively while reading storybooks in different languages, and at the same time effectively tackle the existing limitations of popular electronic book readers.

The key objectives of the project are:

1. Constructing a text recognition model capable of extracting text from coloured pages and identifying the language of the extracted information.
2. Constructing a multilingual speech generation model that can read input text aloud with identified language.
3. Constructing a multilingual speech recognition model capable of translating the vibration of speech into corresponding text.
4. Establishing an intelligent agent that gathers inference from a user's questions and provides correct answers with regard to the visual element in the image.
5. Creating an internal self-labelling system that labels the cartoon image with information gathered from interactions between parents, child and product itself.
6. Testing functionality of each working component and perform integration test after combining to ensure the product works as expected.
7. Designing an aesthetically pleasing visualization interface to show returned output of the product.

2 Related Work

Scholars and corporations have initiated the attempt of applying deep learning in the anime industry. In 2020, IQIYI and the International Joint Conference on Artificial Intelligence (IJCAI) jointly launched the Anime Character Recognition Challenge[6]. On the contrary, AI applied in children story reading is falling behind and barely deployed in the electronic PBR on the market, even though picture books are also cartoon. Hence, the crucial dimension of the project is to explore more AI scenarios in storybook reading, not only for introducing new features but also for a complete upgrade to existing functions.

It should be pointed out that the nature of children's stories is the leading cause of the phenomenon. Unlike anime which appeals to the audience in all age groups, children's storybooks are specialised for young children, which means the content is meaningless to other age groups, even though children will not be interested in this content when they grow up. Thus, the market size of children's storybooks remains small, which also contributes to the limited electronic BRP available to choose.

Product	Type	Characteristics
Luka Reading Robot[7]	Independent Device	• Recognise English and Chinese picture books • Pre-recorded audio for available books • Purchases required for online books
Epic!	Mobile Device App	• Provide high-quality content • English picture books only • Limited audio books available

Table 1: Existing Products

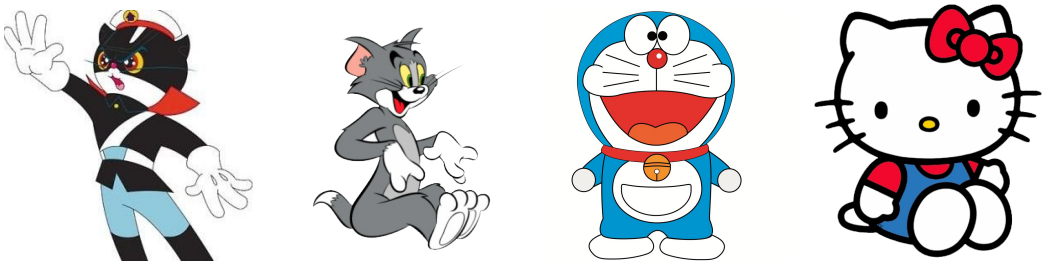
Table 1 lists two PBRs that have the highest market share, along with their type and characteristics. One commonality between these products is that audio for storybooks is pre-recorded by the staff, and only covers a few languages. In this case, despite such products helping

children to read stories, the reading is subject to constraints imposed by the limited number of books and the available languages. Such limitation is resolved in the project with an AI-powered solution, where pre-recorded audio is replaced with deep learning models that perform language detection, text extraction and speech generation.

By doing so, the MIPBR provides greater flexibility that enables parents to upload stories of their own choice, with the sole prerequisites being that the format of storybooks is restricted to PDF. Moreover, the MIPBR puts more emphasis on its multi-lingual capabilities, with an adjustable language setting. By default, it speaks in the language identified from the PDF file and understands voice input in Chinese and English. For the product to understand a user-specified language, the user could download the pre-trained model of that language.

Previously, interactions between children and similar products have been confined to canned responses towards only a few pre-specified questions with regard to the storybook. Thus, one remarkable feature introduced to the new product is the human-AI interaction mechanism, with the main task of answering questions raised by children according to the story. The mechanism integrates a language model and an image recognition model, thus refining questions from voice input and answering them based on the identified information from the storybook. In addition, the mechanism provides voice assistance for children to perform their operation by speaking to the product directly.

However, considering most animes would constantly debut follow-ups while children's storybooks often end up as short stories, the sample size of training data might be insufficient for training the image identification model. Moreover, as can be seen from Fig. 1, different cultures may lead to a distinct artistic style in children's storybooks, the artist's creative style would also exacerbate the significant changes in the appearance of the same thing in different stories, even within the same culture. It is therefore even harder to construct a deep learning neural network on recognising story content with all these biases.



(a) Chinese Cartoon cat (b) American Cartoon Cat (c) Japanese Cartoon Cat 1 (d) Japanese Cartoon Cat 2

Fig. 1: The appearance of cartoon cat by country and artist

The project comes up with the idea of establishing an internal self-labelling system. That is, while encouraging parent-child interaction, the system asks questions proactively about the identification of the cartoon object when the model cannot recognise it, then labels the object with the answer given by the user with their consent. Such a system resolves the immediate problem and collects more training data that can be used for training model with higher accuracy in identifying cartoon objects. Hence, it creates a virtuous circle in the long term, such that the high quality of the product attracts more customers, whereas the increasing customer usage also contributes to the better performance of the built-in models.

In conclusion, once such technologies have been perfected, the business model of MIPBR would upgrade from resource-oriented to user-oriented, thereby remodelling the industry landscape.

3 Functional Specification

In this section, the functions that fulfil the technical demands of the product are enumerated first, followed by an outline of the non-functional requirements, both accompanied by brief descriptions.

3.1 Functional Requirement

1. **Preparedness Functionalities**: First and foremost, given the storybook, the internal components of the MIPBR must be initialised to enable its functionality. Thus, for a specific storybook, the following actions are required:
   (a) **Storybook Uploading**: The user must be able to upload the storybook to the MIPBR, whereas the reader is expected to check the uploaded file type and proceed further only if the storybook is a PDF file.
   (b) **PDF Handling**: The MIPBR must be able to display storybook content on a page-by-page basis, thus the pdf file must be split into images and stored in a temporary folder, with each image named by its page number to facilitate later access.
   (c) **Language Detection**: The MIPBR must be able to recognise the language of the storybook, not only for extracting text, but also for reading in the correct language, in accordance with the multi-lingual capability.
2. **AI Functionalities**: The essential functions of the MIPBR are powered by machine learning models. Considering the distinction in functionality, the MIPBR deployed different models to take advantage of their respective strengths in performing the following tasks:
   (a) **Read Story**: For the MIPBR to deliver content on the colouring page to children by voice, the following steps must be followed precisely:
      i. **Image Pre-process**: The MIPBR is expected to have a reasonable accuracy in identifying text to read, thus noises on the colouring page must be removed as much as possible.
      ii. **Text extraction**: It is mandatory for the MIPBR to extract text from the colouring page, then save it online as a variable and offline into the temporary text file as context.
      iii. **Speech Generation**: Given the identified text, the MIPBR should be able to generate corresponding synthesised speech in the detected language.
   (b) **Answer Question**: One condition for the MIPBR to be interactive is being able to answer questions asked by the user. Thereby the following needs to be accomplished:
      i. **Speech Recognition**: The MIPBR must be able to translate the user's voice input into the corresponding text, and output the text into a temporary file.
      ii. **Natural Language Processing**: Given the text derived from the previous step, natural language processing should be performed on it to filter keyword and determine the task to perform.
      iii. **Object detection**: Since a majority of questions asked by the user are related to the cartoon object on the colouring page, the storybook should be able to answer such questions after performing the image detection.
      iv. **Speech Generation**: Same as above.
   (c) **Label Object**: As mentioned earlier, the storybook will ask questions proactively when it cannot identify a cartoon object, followed by the following actions:
      i. **Seek Permission**: The MIPBR must inform the user about this function, the function is disabled by default without their consent, and only operates once the user agreed.

ii. **Labelling**: The storybook must be able to spot keywords from the answer given by the user that is in accord with the cartoon object as its label.
iii. **Gather Information**: The MIPBR should be able to duplicate and crop the original image to the partition that only contains the object, and name it by its label.

3. **Interactive Functionalities**: Aside from the functions powered by AI, the MIPBR should also provide some basic functions that enable the user to operate the product. Therefore, the following functions exist:

   (a) **User Interface**: The MIPBR must have a user interface that allows the user to see colouring pages when the reader narrates the tale in the storybook, with buttons triggering different functions upon clicking.

   (b) **Execute Instruction**: The MIPBR should be able to execute the user's instruction either by clicking a specific button, or by speaking the instruction directly, e.g. turn to page 11.

3.2 Non-functional Requirement

Nowadays, too much attention has been drawn to functional requirements of a product, whereas non-functional requirements are always overlooked by both developers and users. However, non-functional requirements have a great impact on the availability and stability of the MIPBR. Thus, the project defines the following non-functional requirements:

1. **Simplicity of Operation**: As the MIPBR is in demand as an auxiliary teaching product for early childhood education, and the potential user basically know nothing about computer literacy, e.g. file system. Therefore, the MIPBR should have its functions as friendly and straightforward to operate as possible for the potential user.

2. **Protection of Privacy**: With the increasing involvement of smart devices in people's daily routine, people are facing unprecedented privacy breaches. Even though the protection of privacy has greatly improved recently, information that has been compromised is irreversible. Thus, as a product that is used during childhood for the coming generation, the MIPBR must ensure strict compliance with the privacy protection policies:

   - User must be informed about the self-labelling functionality, and they should be the ones deciding whether this function is enabled.
   - All the data generated during use of the MIPBR must be saved in a temporary folder and destroyed upon closure of the program.
   - With the user's consent, the minimum amount of data associated with the unidentified objects should be collected for training purposes only.

3. **Robustness of System**: Take into account the MIPBR combines multiple models to process voice and image data, as well as the additional need to interact with the user. Thus, it must ensure the robustness of the system to resolve potential conflicts in function invocation by achieving the following:

   - The MIPBR should ensure its stability during use against system failure caused by children's arbitrary operation out of curiosity.
   - A trade-off must exist between the accuracy of model prediction and waiting time for processing data, thus avoiding the distraction of children.

4 Methodologies

This section provides a thorough discussion of the crucial techniques and algorithms used in the project to clarify the reason for the choice, where the order of each sub-section matches the corresponding operation sequence of the MIPBR.

4.1 Data Pre-processing

There are two types of input to the MIPBR: the pdf file selected by the user and the user's speech. For the pdf, it is then split into images and multiple actions are performed on the same image with different focuses. To assure the accuracy of the trained neural networks, images are required to be processed in different ways as per the nature of the specific task. As for the user's speech, it cannot be directly fed into the speech recognition components, thus feature extraction on the speech must be performed.

4.1.1 MFCC Feature Extraction

In the context of ASR, acoustic features are usually based on the spectrum of the signal. Therefore, when training the neural nerwork model, it is preferable to extract acoustics features from the waveform, such as the speech signal, and convert it to the spectral representation to feed the model, rather then use the waveform itself directly.

The Mel-frequency cepstral coefficient (MFCC), one of the most commonly used spectral representations of acoustic features, motivated by the behaviour of human perception sensitivity with respect to frequencies. The MFCC has shown its success in speech recognition, hence, it is used in the project. Below is a step-by-step standard MFCC conversion procedure used in the project[8].

According to the definition of spectral tilt, higher frequencies have less power than the low frequencies.Therefore, amplifying the amount of energy in the high frequencies by passing the waveform through a pre-emphasis filter, the high-frequency part that was suppressed during the sound production mechanism of humans is more available to the acoustic model[9]. Below is the formula of pre-emphasis:

$$y(t) = x(t) - \alpha x(t-1) \qquad 0.95 < \alpha < 0.99 \tag{4.1}$$

After going through the pre-emphasis filter, the waveform is then partitioned into consecutive over-lapping windows. The MFCC of each window can be computed as follows[10]. Firstly, the Discrete Fourier transform (DFT) is applied to the windowed signals. The DFT identifies the amount of energy at each frequency band, and is computed as follows:

$$\hat{X}[k] = \sum_{n=0}^{N-1} e^{-i\frac{2\pi}{N}nk} x[n] \qquad for \quad k = 0, 1, ..., N-1 \tag{4.2}$$

One Mel is defined as one thousandth of the pitch of a 1KHz, but it does not correspond linearly to the physical frequency of the tone. Therefore, Mel should be approximated from physical frequency, the formula is expressed below, where f denotes the physical frequency, and f_{mel} denotes the perceived frequency:

$$f_{mel} = 1127 \times ln(1 + f/700) \tag{4.3}$$

To compute The Mel spectrum, Fourier transformed signal is passed through a set of triangular overlapping filters known as Mel-filter bank, which is implemented in frequency domain. The triangular filter banks with Mel frequency warping is given below:

$$H_m(k) = \begin{cases} 0, & k < f(m-1)) \\ \frac{2(k-f(m-1)))}{f(m)-f(m-1)))}, & f(m-1) \le k \le f(m)) \\ \frac{2(f(m+1)-k)}{f(m+1)-f(m)}, & f(m) < k \le f(m+1) \\ 0, & k > f(m+1) \end{cases} \tag{4.4}$$

Then, the log-energy at the output of each triangular Mel weighting filter is computed as follows:

$$S[m] = \ln\left[\sum_{k=0}^{N-1} |X_a[k]|^2 H_m[k]\right] \quad 0 \le m < M \tag{4.5}$$

Finally, the cepstrum is obtained by computing the discrete cosine transform as follows: The dimension of a complete MFCC feature vector varies based on the length of the signal and the number of frames it partitions to.

$$c[n] = \sum_{m=0}^{M-1} S[m] \cos \pi n(m+1/2)/M \qquad 0 \leq n < M \tag{4.6}$$

It is essential to mention the detailed description of MFCC feature extraction here because the function `mfcc` from Python module `python_speech_features` only accepts a frame rate of 16KHz and FFT size of 512. Otherwise, the frame will be truncated, which would result in an unexpected situation. Therefore, the `mfcc` function is re-implemented in `mfcc_base.py` following the above steps, to make it feasible with larger frame rate (48KHz).

4.1.2 Otsu's Thresholding

In most cases, two actions are performed on a single image: text extraction and object identification, with the nature of these two actions corresponding precisely to the foreground text and background cartoon. In this case, the image could be considered equivalent to data with two classes, and a threshold parameter should be therefore specified to distinguish foreground text from background cartoon.

In image thresholding, input to the algorithm is usually a grayscale image and a threshold value. Since colouring pages compose the contemporary storybook, the split image needs to be converted from RGB into grayscale first. As for the threshold value, it is specified manually by analysing the pixel intensities obtained from the image in the conventional approach.

With the image statistics and the knowledge above, image thresholding is achieved by comparing the specified threshold value to the global pixel intensities. That is, a pixel is classified as foreground if its intensity is greater than the threshold and background vice versa. However, considering the various artistic style of children's storybook, a fixed threshold value cannot handle all the images.

Otsu's Thresholding is then introduced to find the optimal threshold value of the input image adaptively by iterating over all possible threshold values (from 0 to 255). As described by N.Ostu in his paper[11], the input grayscale image is transformed into a histogram of the distribution of pixels first and foremost, as shown in the demo Fig. 2 below:

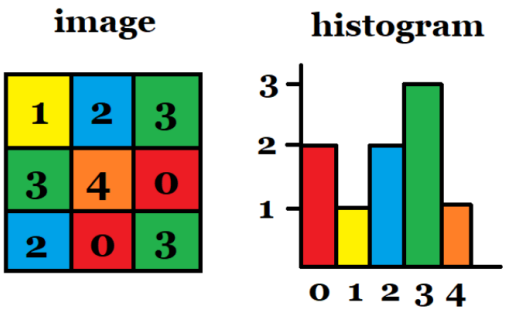


Fig. 2: Example of image to histogram transformation

The core idea of Otsu's Thresholding is to separate the histogram into two clusters by defining a threshold that minimises the weighted variance of these classes denoted by $\sigma_w^2(t)$. The whole computation is defined by the expression below:

$$\sigma_w^t = \omega_1(t)\sigma_1^2(t) + \omega_2(t)\sigma_2^2(t) \quad where \quad 0 \leq t \leq 255 \tag{4.7}$$

To be more specific, when transforming the input image to a histogram, a probability function P is generated for each pixel value. Then, normalisation is performed on the resulting distribution to ensure it follows a probability distribution. After that, the distribution is divided into two clusters C_1 and C_2 by a threshold of value t, using the class distribution probability denoted by $w_1(t)$ and $w_2(t)$ in the equation, defined as follows:

$$\omega_1(t) = \sum_{i=1}^{t} P(i) \qquad \omega_2(t) = \sum_{i=t+1}^{255} P(i) \tag{4.8}$$

With the pixel intensity of C_1 lies within the interval [1,t] and C_2 within [t+1,255], the mean for each class is obtained:

$$\mu_1(t) = \sum_{i=1}^{t} \frac{iP(i)}{\omega_1(t)} \qquad \mu_2(t) = \sum_{i=t+1}^{255} \frac{iP(i)}{\omega_2(t)} \tag{4.9}$$

Actually, N. Ostu mentioned two means of finding the optimum threshold value in his paper, either by maximising the between-class variation or minimising the within-class variation, whereas the method mentioned earlier is by minimising the within-class variation. The expression for maximising the between-class variation is also given as follows:

$$\sigma_b^2(t) = w_1(t)w_2(t)[\mu_1(t) - \mu_2(t)]^2 \quad where \quad 0 \leq t \leq 255 \tag{4.10}$$

4.2 Speech and Language Processing

Neural network has a long history of usage in the field of speech recognition, with the input speech waveform as a typical sequence data that comes with a particular order. In most case, a neural network processes its input independently, which means the previous input has nothing to do with the current input. In this case, the neural network could have a decent performance in word-level transcription, but readers may also find it difficult to understand when these words are connected as a whole. To resolve the issue, the project takes advantages of the following artificial intelligence techniques.

4.2.1 Recurrent Neural Network

As is widely acknowledged that speech is coherent deep in time, remembering the pronunciation of the current timestamp is in such a way helpful for predicting the incoming pronunciations. Recurrent Neural Network (RNN), as an effective neural network architecture for dealing with sequential learning tasks, is naturally of great facilitates for speech recognition where the previous input is highly correlated with the subsequent.

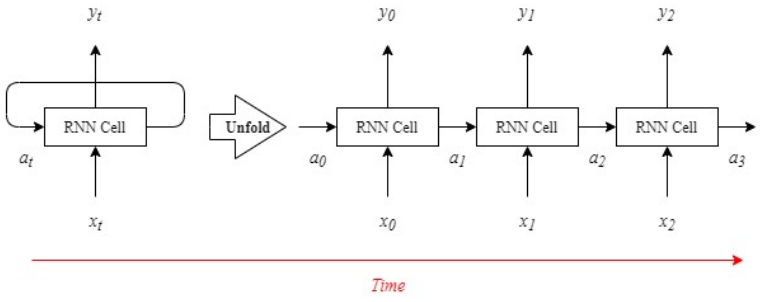


Fig. 3: Partial RNN Structure

As shown in Fig. 3, a typical RNN model can be considered as the repetition of a single RNN cell, which receives data outside the neural network from input layer denoted by x_t, performs computation involving the activation a_t inside the RNN cell, and produces the output y_t. One

of the most important characteristics of RNN is that historical information is taken into consideration, as the weight of the current RNN cell is passed to the next RNN cell across time through recurrent connections, thus maintains contextual information in memory and provide further robustness to warping along time than non-recursive models[12].

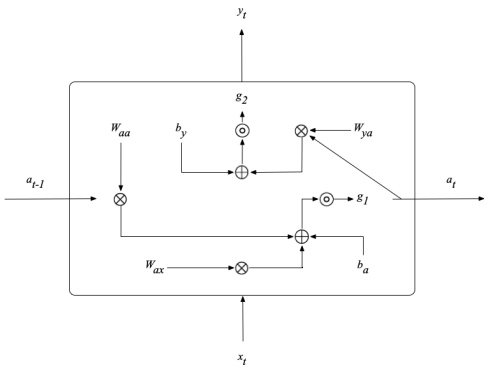


Fig. 4: RNN Cell Internel Computation

Look more closely at the RNN cell, it is the hidden layer of the neural network that defines the state space of the whole system, with $W_{ax}, W_{aa}, W_{ya}, b_a$ and b_y being the temporal coefficients shared by all the cells, and g_1, g_2 as two activation functions[13]. Fig. 4 depicts the detailed computation using these variables and functions, where formula for the activation a_t and the output t_t in given timestamp t is expressed in Equation 4.11 and 4.12 respectively:

$$a_t = g_1(W_{aa}a_{t-1} + W_{ax}x_t + b_a) \tag{4.11}$$

$$y_t = g_2(W_{ya}a_t + b_y) \tag{4.12}$$

Theoretically, this architecture enables RNN neurons to learn from the input and passes context to the next timestamp, thus builds a capacity for RNN to save, remember and process complex temporal signals on a long-term basis. However, just like other standard neural networks, the standard RNN suffers significantly from the Vanishing Gradient Problem[1] and the Exploding Gradient Problem[2], which means the context could be greatly deluded over a long distance. Therefore, It is impractical to see how meanings in faraway words dictate overall meaning. For example, given the following sentences, it is very likely for the RNN model to predict the word "sky" in the first sentence since the context is not far away. Nevertheless, a vanilla RNN cannot ensure accuracy when predicting the word "Chinese" in the second sentence:

1. "The fish are in the **sea**"
2. "I moved to China since ..... I can speak fluent **Chinese**"

4.2.2 Long Short-term Memory (LSTM)

To resistant to the vanishing and exploding gradient problem, several variants of RNNs have been proposed by researchers. The LSTM architecture, among all the variants, is widely considered the most successful variant[14].

Just like vanilla RNN shown in Fig. 3, a typical LSTM is also formed with recurrently connected components, with the difference being the component itself changed from RNN cell to LSTM

[1]When training a gradient-based neural network, small error gradients accumulate and result in very small updates to neural network model weights during training, thus effectively prevent the network from learning from data.

[2]On the other hand, large error gradients accumulate and result in very large updates to neural network model weights during training, thus effectively prevent the network from learning from data.

memory block. Each memory block is composed of a self-connected memory cell and three gates: input gate, output gate and forget gate.

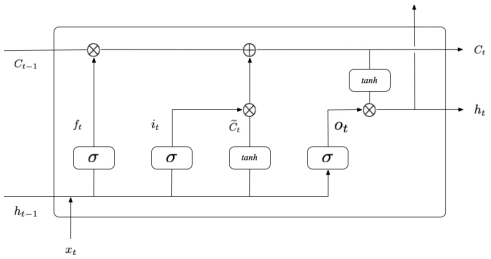


Fig. 5: LSTM Memory Block

Fig.5 provides a deep insight into the structure of a LSTM memory block, where the input gate, output gate and forget gate in the given timestamp t is denoted by i_t, o_t and f_t, and σ represents the logistic sigmoid function. According to the structure of LSTM memory block in Fig. 5, its internal computation can be implemented by the following composite formulas, where W_* and b_* are the corresponding weights and biases:

$$\begin{aligned} f_t &= \sigma\left(W_{xf}x_t + W_{hf}h_{t-1} + W_{cf}c_{t-1} + b_f\right) \\ i_t &= \sigma\left(W_{xi}x_t + W_{hi}h_{t-1} + W_{ci}c_{t-1} + b_i\right) \\ \widetilde{c}_t &= \tanh(W_{x\widetilde{c}}x_t + W_{x\widetilde{c}}h_{t-1} + b_{\widetilde{c}}) \\ o_t &= \sigma\left(W_{xo}x_t + W_{ho}h_{t-1} + W_{co}c_t + b_o\right) \\ c_t &= f_t c_{t-1} + i_t\widetilde{c}_t \\ h_t &= o_t \tanh\left(c_t\right) \end{aligned} \tag{4.13}$$

The first step in the LSTM memory block is to decide whether the previous memory should be discarded, which is performed by the forget gate. To be specific, output of the forget gate is either 1 or 0, memory is to forget when the computation output of forget gate is 0 and is preserved vice versa. At the same time, the input gate controls whether the current input to the LSTM memory block is considered. On the other side, taking the previous cell value into consideration, a vector of new candidate $\widetilde{c}_t$ is created, together with the output of forget and input gate to calculate the new cell state c_t. Finally, memory pass thought the output gate is transferred to hidden state.

Such architecture enables the LSTM to control the flow of signals, in such a way that useful information is remained in memory while insignificant one is discarded. Consequently, long term dependencies are established, and the vanishing and exploding gradient issue is mitigated significantly.

4.2.3 Bidirectional Long Short-term Memory (Bi-LSTM)

The shared disadvantage of the RNN and its variant LSTM is that only previous context is accessible, but it is clear that later words in a sentence also provide context to earlier ones. Thus, it is beneficial to have a neural network with look-ahead ability. That is, the network is able to access the past context, as well as the future context, thus it could learn the semantics of the sentence more accurately.

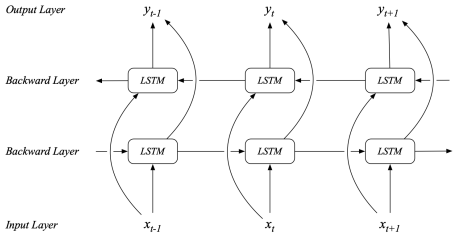


Fig. 6: Bidirectional LSTM

As is shown in Fig 6, Bi-LSTM is literally the concatenation of two LSTM architectures, which extends the uni-directional LSTM by introducing a second hidden layer[15]. The forward layer, as in the original LSTM architecture, processes the past context of the input sequence before given timestamp t, whereas the backward layer processes the input sequence after t in a reverse order. Thus, in each time step, the neural network is able to exploit the past and future information.

4.2.4 Connectionist Temporal Classification (CTC)

The aforementioned speech recognition techniques are based on phonetics. That is, giving the training audio clips, the speech recognition model should learn how to predict their transcripts based on the intermediate representation of extracted audio signals in terms of phonetic symbols. However, it has no idea about how the characters in the transcript align with the audio signal, e.g. which word corresponds to which voice in the audio clip.

In this context, the corpus of the training model requires each character to be manually aligned to its location in the audio, which is prohibitively time-consuming for any reasonably sized dataset. On the other hand, the fixed rule derived from the training dataset might not be feasible with unseen data due to the fact that people's speaking speed vary. In the light of these considerations, the CTC approach is introduced to the project.

CTC is a type of neural network output and associated scoring function for training an artificial neural network (BiLSTM in our case) to solve the sequence labelling issue where alignment between the input acoustic signal and the target label is unknown or incorrect. It first performs computation on the raw output from the last layer of the Bi-LSTM model, with its result high for a single acoustic signal to indicate the presence of the symbol, whereas others are labels "blank". In this way, the duration information is discarded, and tedious manual labelling is abandoned, making the training process more straightforward. Moreover, for the scoring function, it calculate probability on multiple sequences, where a sequence is a set of all possible texts in the speech data. Then compare the predicted result with the label, calculate the error, and continuously update the network weight[16].

There are two tasks in the project that are associated with CTC: calculation of the loss value to train the neural network, and decoding of the output of the neural network to get the text in the speech model and the text recognition model.

CTC Loss Function The loss function formula is expressed in Equa. 4.14, where x and l denote the input sequence of the acoustic signal and labelling transcripts, and z^l is the latitude encoding all the possible alignment between x and l that allows the existence of label repetition interleaved with blank labels[17].

$$\mathcal{L}_{CTC} = -\sum_{(\boldsymbol{x},l)} \ln p\left(\boldsymbol{z}^l \mid \boldsymbol{x}\right) \tag{4.14}$$

The computation for correct labelling probability $p(\boldsymbol{z}^l \mid \boldsymbol{x})$ from Equa. 4.14 is expressed in Equa. 4.15, which uses forward variable α and backward variable β to represent the summed

probability from the initial state at all paths in z^l to the target state u in the given timestamp t in both directions.

$$p\left(\boldsymbol{z}^l \mid \boldsymbol{x}\right) = \sum_{u=1}^{|\boldsymbol{z}^l|} \alpha_{x,z^l}(t,u)\beta_{x,z^l}(t,u) \tag{4.15}$$

CTC Beam Search Decoding CTC uses standard beam search algorithm for decoding. To e specific, the beam search algorithm will traverse the output of the Bi-LSTM model, and use an external language model to build text candidates (Beams) and calculates the score for each candidate. Then it turns to decode the most likely output sequence after searching through all the possible output sequences based on their score returned by the language model.

The pseudo-code for the algorithm is shown above. At first, a list of beams is initialised with an empty beam and a corresponding score(line 1-2). Then step into the loop, only the beam with the highest score is kept at each loop (line 4-5). Further, the best beam is expanded by all possible characters from the word dictionary literately, and applied to the language model. After the last loop, the best beam selected from the list with a length of beam width (BW) is returned.

Algorithm 1: CTC Beam Search Algorithm with Language Model (LM)[18]

Input : Acoustic Model Output matrix mat, BM and LM
Output: Decoded Transcript

1 $beams$ = $\{\emptyset\}$
2 $P_b(\emptyset, 0)$ = 1
3 **for** $t \leftarrow 1$ **to** T **do**
4 $bestBeams$ = $findBestBeams(beams, BW)$
5 $beams = \{\}$
6 **for** *b* $\in$ *bestBeams* **do**
7 **if** *b* != $\emptyset$ **then**
8 $P_{nb}(b,t)$ += $P_{nb}(b,t-1) \cdot mat(b(-1),t)$
9 **end**
10 $P_b(b,t)$ += $P_{tot}(b,t-1) \cdot mat(blank,t)$
11 $beams = beams \cup b$
12 **for** *c* $\in$ *alphabet* **do**
13 $b' = b + c$
14 $P_{txt}(b') = applyLM(LM,b,c)$
15 **if** $b(t) == c$ **then**
16 $P_{nb}(b',t)$ += $P_b(b,t-1) \cdot mat(c,t)$
17 **else**
18 $P_{nb}(b',t)$ += $P_{tot}(b,t-1) \cdot mat(c,t)$
19 **end**
20 $beams = beams \cup b'$
21 **end**
22 **end**
23 **end**
24 **return** $findBestBeams(beams, 1)$

4.3 Image Object Detection

Compared to the text embedded in the colouring page of the storybook, the cartoon objects are more attractive to pre-school children and more straightforward for them to understand the storybook content. As declared above, the MIPBR does not require pre-existing information for any storybook, the need to recognise objects from the colouring page is fulfilled by the AI-powered object detection techniques.

4.3.1 Convolutional Neural Network(CNN)

In regular neural networks, one neuron in the hidden layer has learnable weights and biases, and is connected with all the neurons from the previous layers, thus forms the fully connected architecture, where the last layer being the output layer that produces output of the network. However, image is high dimensional as each pixel is considered a feature, resulting in the expensive computation for calculating and tuning the biases and weights for each neuron, especially when the project could not take advantage of pre-trained models as there is no such model for children's storybook. In this case, the CNN with the dimensionality reduction feature proposed by Krizhevsky et al.[19] naturally draws the attention of the project.

A study conducted by Razavian et al.[20] mounted that CNN has been proven an effective category of neural network for computer vision related tasks, such as image object detection. Unlike a regular neural network that takes vectors as input, CNN assumes its input to be multi-channel images, in our case 3 channels for the input RGB colouring images from the storybook. Then, by stacking different layers together and exploiting their respective advantages, the number of parameters could be effectively reduced without declining the quality of the model. Fig. 7 depicts a typical CNN architecture, which is composed of a sequence of hidden layers: the convolution layers (CONV), the pooling layers (POOL) and the fully connected layers (FC), with the volume of activation in the current layer transforming to its following layer via a differentiable function.

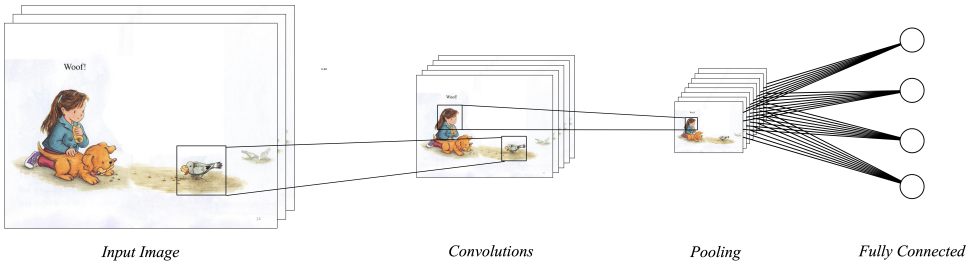


Fig. 7: Overview of CNN Architecture

As indicated in the model's name, CONV is the core component in the convolutional network where most computational heavy lifting is performed in this layer. It works by defining first a filter that only receives a sub-region of the input raw image, then performing convolution operation on the filtered region, and producing the final extracted features from the region. By repeating this procedure until the entire image is scanned, CONV derives a feature map from the raw image.

Likewise in POOL, filters are used to perform downsampling operation on the feature map derived from the previous layer along its spatial dimensions. For example, using the most common max-pooling as the downsampling operation, only the maximum value in the filter projected subarea is retained and forward to the subsequent layer. Therefore, the amount of parameters from the input image is heavily reduced after going through CONV and POOL, and is more representative, which aids to the mitigation of training cost as well as the risk of overfitting the model.

With the massive amount of computing being relatively eased, CNN naturally inherits the fully connected architecture from neural network as its last layer. In FC, the weight and bias of each neuron is trained and tuned via gradient descent with backpropagation, and its output is the optimised class score, which represents how confident CNN is to classify the object in the image in consistent with the given label.

4.3.2 YOLO (You Only Look Once)

Despite the aforementioned successes, traditional CNN has obvious limitations as an early proposed convolution-based neural network for image classification. Namely, though traditional CNN can determine whether the object of interest is present in the image, it cannot deal with storybook pages with multiple objects properly, as well as indicate where each object locates. Moreover, the huge amount of training data for CNN to achieve sufficient accuracy is not available in the project.

In the light of the above considerations, YOLO is introduced to the project as the approach of image object detection. YOLO was initially proposed by Redmon et al.[21] in 2015, with a refreshing simple architecture shown in Fig. 8. Namely, YOLO is composed of 24 CONV and 2 FC, where the layers function the same as in ordinary CNN, CONV for feature extraction while FC for prediction.

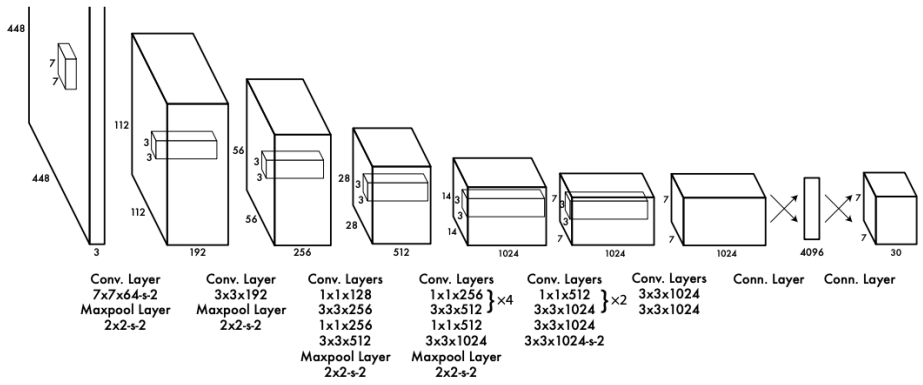


Fig. 8: Architecture of YOLO[21]

One reason for choosing YOLO in the project is that, YOLO has extremely fast speed in training and testing without compromising the prediction accuracy. Just like the name suggests, YOLO is fast because it only requires one forward propagation through the neural network to make predictions, which effectively reduces the waiting time for the user to get response. On the other hand, tremendous progress has taken place in YOLO, 5 version of YOLO were developed over the past few year, with the latest one being the current state-of-the-art.

5 Design

This section provides an overview of the system design. To be specific, it contains a high-level overview of the MIPBR architecture, the workflow of the proposed components, as well as how these components are integrated into the MIPBR with innovative design concepts.

5.1 Overview

Fig.9 elaborates the details of the MIPBR architecture, which involves three layers from left to right: user, interface and backend. The interface comprises two scenes: the start scene and the main scene, whereas the start scene is displayed initially when the program starts. The user should be able to input an external file into the start scene, the middle layer interface would then validate the input file and proceed further when the input file is in pdf format.

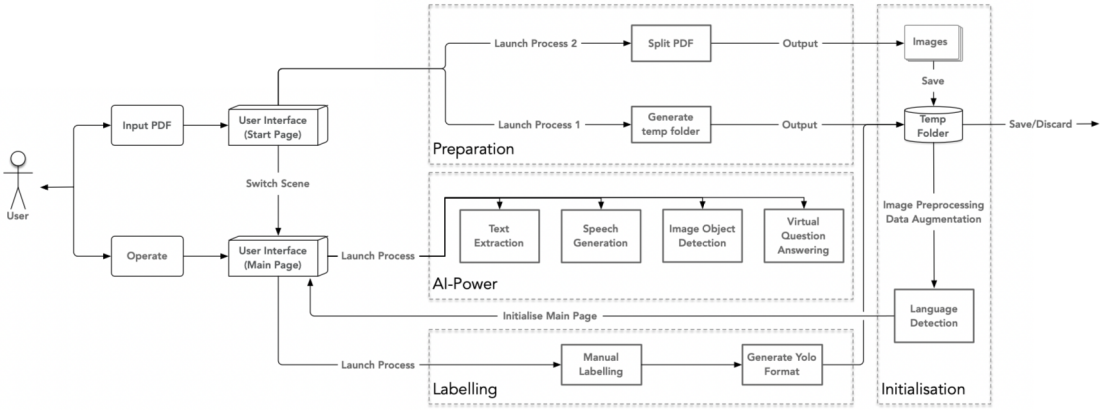


Fig. 9: Overview of the MIPBR

As shown, the backend tasks in the right layer are all wrapped in processes and categorised by their nature and stage. Among the four partitions, preparation and initialisation are the stages where tasks are auto-triggered sequentially to process the pdf file received and initialise the main scene, followed by the automatic switch from the start scene to the main scene.

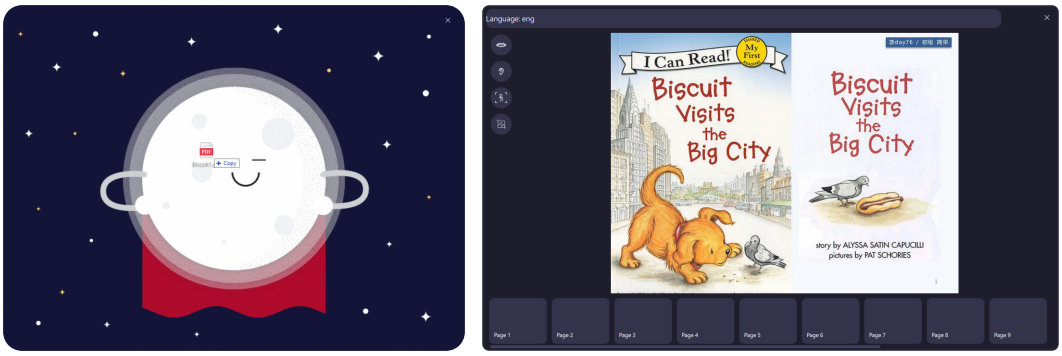
The main scene presents the input pdf file to the user by loading the split image, with buttons aside that allow the user to click for triggering functions and operating on the MIPBR. The functions fall into two categories: functions empowered by deep learning models, and the object labelling tool. Moreover, these functions are mutually exclusive, which means the user cannot trigger two different functions at the same time, as well as trigger the same function before the previous work process is finished or killed, detailed design of the functions are available in sections concerning individual components.

The MIPBR produces data during its runtime. Namely, to facilitate access to the images split from the pdf file, it creates a temporary folder to store these images, as well as files generated by the object labeling tool. Regarding user privacy, the user would be informed clearly about the purpose of the temporary folder, and the folder would be deleted by default when the program shuts down. Data would only be collected with user consent and for training purposes only.

5.2 User Interface

Considering time required for the pdf to go through preparation stage grows linearly with the storybook size, plus the additional time required for the storybook to identify the storybook language, the user needs to wait the relatively long time of approximately 30 seconds. In this case, the UI is designed to have two scenes as demonstrated in Fig. 10, where the start scene for taking user input has occasional animated background to draw the children's attention when the main scene is waiting for the setup value generated in the initialisation phase. In addition, both scenes are in dark mode to protect the children's eyesight from possible eye fatigue, especially when children tend to read the storybooks before sleeping.
Due to the fact that the MIPBR is designed for pre-school children who are assumed to have little or none computer related knowledge, the tradition approach of file upload through the file explorer is not feasible and user-friendly. The approach taken by the project allows the user to upload file by dragging it directly to the start scene, thus children are able to operate the MIPBR independently with the sole prerequisites being that parents place the pdf file prominently, e.g. on the desktop.

As for the main scene, it is designed as a pdf viewer to display the content of the pdf file. The main body of the scene is the image retrieved from the temporary folder. The text on the top bar indicates the language of the file, and the bottom slidable panel enables the user to



(a) Start Page (b) Main Page

Fig. 10: Two Scenes of the UI

flip through the storybook virtually, where each panel has a index to represent its page number. The buttons on the left bar are the services provided by the MIPBR, including reading the story, listening to the user, detecting the object in the image, and invoking the labelling tool.

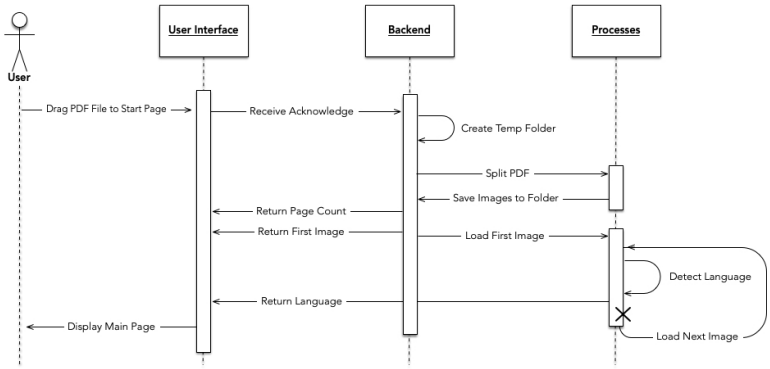


Fig. 11: Workflow of Switch Scene

Fig. 11 specifies the underlying workflow from the user uploading the pdf file to the program switching to its main scene. As soon as the UI receives the pdf file, the backend would create an empty temporary folder, followed by launching a process to split the pdf into images and store them into the folder. and return the page count and first image as setup value. The UI has an internal repeater to create the required number of panels according to the page count, and an adapter to fit the first image into the content window. Meanwhile, the backend would launch another process to feed the first image to the language detection model. If the model works properly, the detected language would be sent directly to the UI as the final setup value, otherwise the process would visit the temporary folder to fetch the subsequent image and follow the same procedure. Cases resulting in language detection failures and how they are addressed is discussed in the problem encountered section.

5.3 AI Empowered Functions

According to the project objectives, the MIPBR should be able to read stories in the identified language, as well as answer the user's questions in accordance with the text extracted and the objects detected. These tasks are achievable via taking the advantage of the deep learning, and each task integrates multiple models to perform its work.

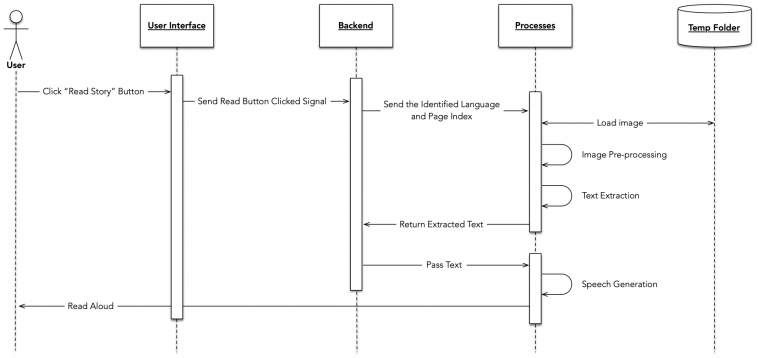


Fig. 12: Workflow of "Read Story"

Fig. 12 describes the workflow from the user clicking the "read story" button to the MIPBR start reading story in the image, whereas the waiting time is mitigated thanks to the previous language detection and the temporary folder that stores the split image. As the backend already knows the language and the current page index, it is able to launch a process immediately upon receiving the signal from the UI. The process would visit the temporary folder to grab the specified image, followed by pre-processing the image, which brings better performance for the text extraction model. The extracted text is sent to the backend at the completion of the process. After that, the backend launches another process, which feeds the text and language option to the speech generation model, thereby reading the story.

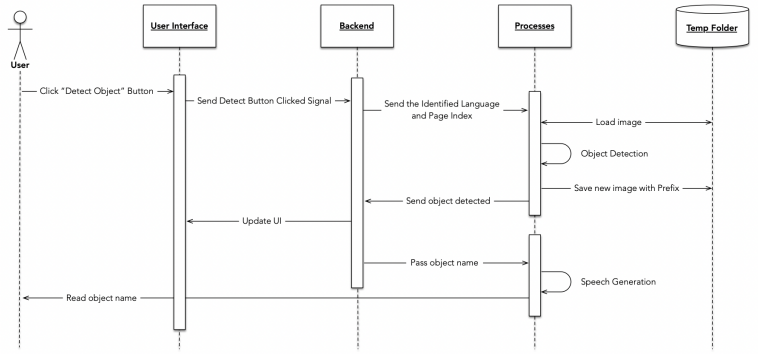


Fig. 13: Workflow of "Detect Object"

As shown in Fig 13, the workflow for object detection is very similar to the aforementioned storybook reading, but with the exact opposite interest. In storybook reading, image objects are usually neglected to avoid disrupting text extraction. In contrast, image objects are the subject to be identified in object detection. In this case, pre-processing of the image is abandoned to retain information delivered by the image and increase the detection accuracy.

Following this procedure, the backend would launch a process to fetch the image displayed on the screen from the temporary folder, then perform the object detection immediately. During this time, the model would duplicate the image and add bounding boxes to represent the objects detected, and this image is then saved to the temporary folder. Meanwhile, the backend would refresh the screen with the new image and launch another process to inform the user of the object identified audibly.

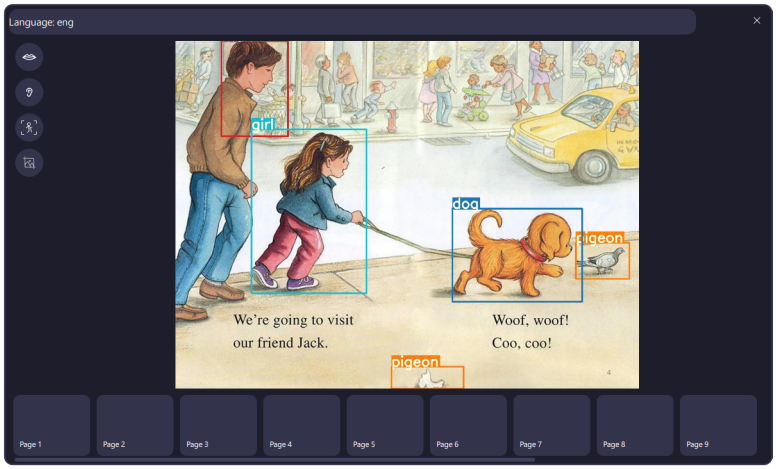
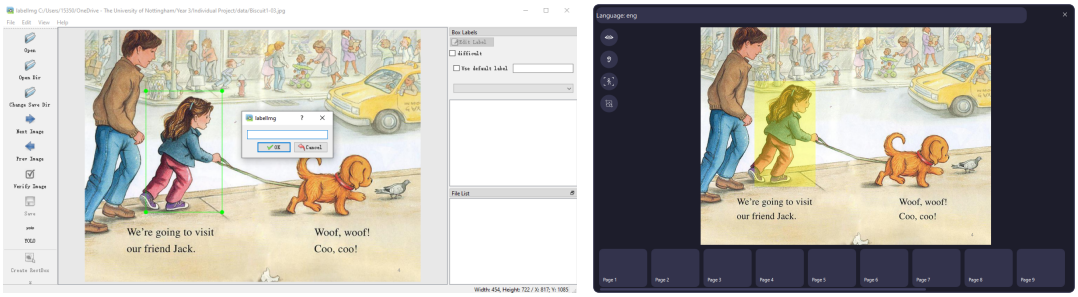


Fig. 14: Refresh Screen

5.4 Labelling Tool

The idea of having a built-in labelling tool comes from the fact that the application of AI is still in its start-up phase in storybook reading, where more training data is desired to contribute to its upgrowth. While the design of the labelling tool is inspired by the existing tool used in the project to label the object for training.



(a) Labelimg (b) Built-in Labelling Tool

Fig. 15: Comparison between Labelling Tools

Fig. 15(a) is a screenshot of a popular labelling tool *labelimg*, which includes the basic functions for uploading file (either single image or folder with multiple images), selecting the rectangle area surrounding the object, labelling the object in the popup window, and producing annotation file in the user-specified format, these functions are all inherited to the built-in tool with adjustment to facilitate children.

As depicted in Fig. 15(b), the built-in labelling tool is specialised for pdf files. Originally, if the user tends to label object in the pdf file using *labelimg*, they need to split it into images first using another tool, then upload the folder containing those images, whereas all these steps are assembled and automated in the MIPBR. Moreover, considering that children are not very good at typing, the pop-up window for typing labels is replaced by a voice recognition model that integrates name entity recognition. Lastly, an annotation text file is produced in the temporary folder for each image that performs the object detection action. When shutting down the program, a YAML file is also produced with folder path and label classes.

Suppose that the user is willing to share their data for training models. In that case, the temporary folder is designed to be eligible for putting into training directly without any further

manual participation. Nevertheless, unlike in *labelimg* where the image size is maintained, the storybook UI would scale the image to fit its UI. As is shown in Fig. 16, since the width of the image is greater than its height, the image is centered and scaled-down till it reaches the same height as the UI, whereas the remaining part of the UI is blank. Follow this arrangement, when the UI records the mouse event, the coordinates being sent back to the backend includes the width of the blank space on the left, which cannot be simply removed as the width varies according to the image size. Therefore, the previous adapting procedure is mathematically reversed in the backend to generate the output correctly, the detailed steps are mentioned in the implementation section.

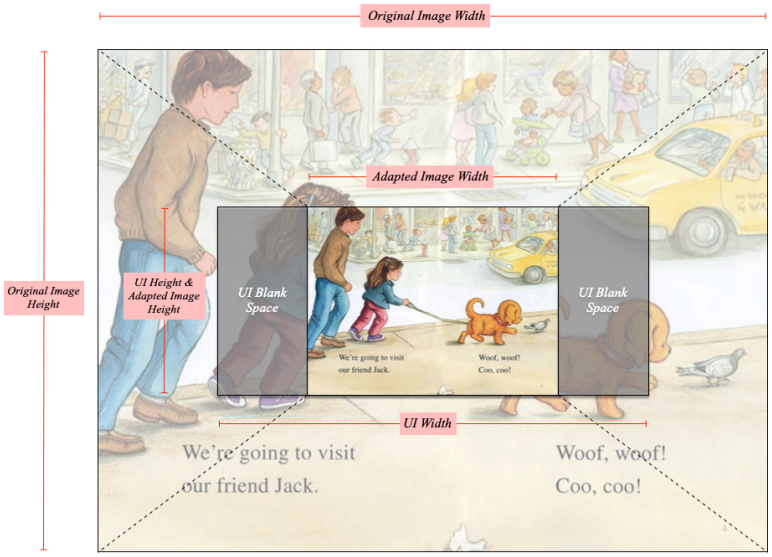


Fig. 16: Adapt Image to UI

6 Implementation

This section elaborate the details of the prototype implementation, including data for training the deep learning models, system implementation for the UI application, implementation for deep learning models, and problem encountered with the way they are solved.

6.1 Data

Two types of data are obtained for training the speech recognition model and object detection model, and is stated as follows.

6.1.1 Audio Data

The Chinese speech corpus used for building the experimental speech recognition model is THCHS-30[22] released by Tsinghua University. The corpus itself contains more than 30 hours of speeches from young university students who speak standard Mandarin in 16,000 Hz `.wav` format, and the corresponding transcripts. The dataset is further split into two parts, where the training set contains 10,000 samples, and the testing set contains training contains 2,495 samples.

6.1.2 Image Data

As discussed in the related work, there is no appropriate cross-culture dataset with storybook cartoon characters. Therefore, the project decides to build its own custom dataset for training the experimental image object detection model. Images in the dataset are obtained from four

storybooks in distinct styles, and are labeled with corresponding image object classes manually. Moreover, one text file is generated per image, containing object annotation as specified by the YOLO model training thread.

YOLO Format Follow this format, each row in the text file represents one object, starts with the index of label as specified in the YAML file, and followed by its x-centroid, y-centroid, width and height (the "Bounding Box") in the image. These annotations are not simply the centroid coordinate or the side length of the bounding boxes, these are rather relative to the width and height of image.

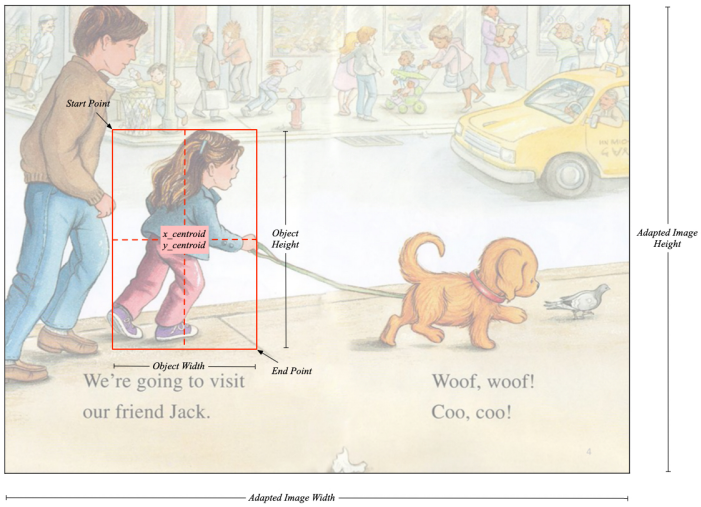


Fig. 17: Formation of Annotation

As depicts in Fig. 17, the red rectangle is the area selected by the user, the start point is the mouse position when the user presses and holds the button, the end point is the mouse position when the user release the button. Its annotation is expressed as follows:

$$\frac{|x\_end - x\_start|}{2 \cdot ImageWidth} \quad \frac{|y\_end - y\_start|}{2 \cdot ImageHeight} \quad \frac{|x\_end - x\_start|}{ImageWidth} \quad \frac{|y\_end - y\_start|}{ImageHeight}$$

6.2 Programming Language

The backend implementation is entirely written in Python 3.7. Python is an interpreted high-level programming language[23], which makes use of a large variety of productive external libraries and modules.

The user interface is written in Qt Modeling Language(QML), a declarative language that describes how the user interface looks and how it behaves. The project chose QML because it provides more flexibility and modernity among all the GUI frameworks (or toolkits) available for Python.

The following list presents brief descriptions of the crucial modules or libraries used, along with what they will be used for:

- `TensorFlow-gpu`[24]: a Python end-to-end open source platform for machine learning, containing a large ecosystem of tools and libraries etc. The gpu version is used in the project to build the BiLSTM acoustic model with gpu devices accelerating the training process.

- `PyTorch`: Another open-source machine learning module used in the project to build the YOLO object detection model, also provides support to the pre-trained language models for natural language processing.
- `PySide2`: The official Python module from the Qt that provides complete access to the Qt framework. It is used in the project to build a modern UI and achieve communications between Python and QML.
- `gTTS`[25]: a Python package and command-line interface tool to invoke Google translate API. Provides multilingual support and speech generation, which will be used for reading aloud the extracted text in the project.
- `Pytesseract`[26]: Python-tesseract is an OCR tool for Python, which wrappers Google's teseract-ocr API. It provides support for all image types and over 100 languages. It is used in the project to calculate confidence score for each language in a user-specified language list, as well as recognise and extract text embedded in the image.
- `OpenCV`[27]: a Python package specialised for computer vision, machine learning and image processing. It is used in the project to process image to manipulate raw images, as well as retrieve data from images.
- `Matplotlib`: a Python package used in the project as a visualisation tool, which visualises performances or outputs of the deep learning models as graphs for demonstration.

6.3 System Implementation

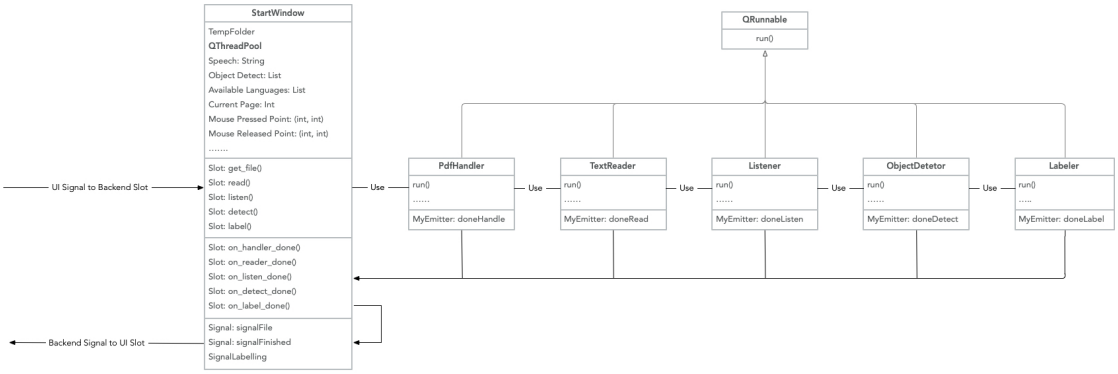


Fig. 18: Class Diagram of *main.py*

6.3.1 Signal Slot Communication

Since the UI and the backend are written in two different scripting languages, a communication mechanism is required to be established between backend and UI objects to trigger functions and pass variable values. QML introduced its signal and slot mechanism to substitute the traditional callback method, where signals are emitted when a particular event occurs, this could be either the user clicks a UI button or the backend task is completed, while the slot is a function that is called in response to a particular signal. Connections are formed by linking signals to slots, such link could be one to one or one to multiple.

On the UI side, QML has a *connections* property, where the backend is specified as target and is declared as a QML global variable that is able to emit signals within any QML properties. Moreover, within the *connections* block, QML local variables are waiting to be assigned with the value passed by the backend signal, then update the UI as soon as it receives signal from the backend.

On the backend side, the *Start Window* object has two clusters of slots, where the first slot cluster is responsible for receiving external UI signals, the second slot cluster is responsible for receiving internal signals emitted by the worker thread. To be specific, as soon as the UI signals reach the backend, the slot function is called. As depicts in Fig. 18, the slot function would pass the torch on to a particular worker thread, and the work thread would exploit the emitter from *MyEmitter* class to pass the result to the second slot cluster. Again, the slot function is called, but this time the work in performed by the backend, and is meant to update the backend data. Lastly, the backend emits signals to the UI objects, resulting in the UI component being refreshed by the value wrapped in the corresponding signal.

6.3.2 Multi-threading

During the integration testing stage, it was found that the UI animation freezes when the start scene receives the pdf file and the backend starts processing it. The same thing happens again when the backend launches a thread to read the image content. During this period, the project tests the UI by emulating the user behaviour and found that the frozen UI is very likely to end up with a crash if the user kept clicking on the screen.

In view of the above, multi-threading is introduced to the project to tackle the issue. There are many methods empowered by the QT to achieve multi-threading, in this project we use the *QRunnable* class. As is shown in Fig. 18, functions of the MIPBR are decomposed to the classes that implement the *QRunnable* interface, these functions are therefore embedded in the *QRunnable* Objects.

The *QRunnable* objects are managed by the *QThreadPool* class. More specifically, when the slot function is triggered, it appends the corresponding *QRunnable* object to the thread pool, the thread pool would then launch a worker thread to execute code in the *run* function. Followed by the *run* function pointing to other functions in the object and so on. The last function performed by the object is always to emit the thread output to the backend. Once completed, the thread pool would recycle the *QRunnable* object. By doing so, tasks are able to run independently without freezing the whole program.

6.3.3 Model View Presenter (MVP)

The project encompasses a wide range of techniques, and the combination of these techniques has greatly complicated the structure of the prototype implementation, especially when they are interrelated with each other. Given this situation, the project follows the MVP design pattern to separate the concerns of visualising, processing and coordinating for its prototype implementation.

By definition, model is a set of classes that defines the application's logic and how the data is processed and manipulated with respect to the task. As implied in Fig. 18, models in the project is the set of classes that implements the *QRunnable* interface.

In this project, view is the user interface of the application, which allows user to input files and perform operation. Due to the fact that the user interface is implemented using QML, it requires no special change to the implementation and is naturally separated.

According to the definition of the presenter, it receives user input from the view, processes the input with help from the models, and pass the result back to the view. In this case, the project considers the *StartWindow* class as the presenter that schedules the worker threads, communicates the worker threads and the UI, and stores the data emitted by signals. Furthermore, the backend processes the data passed into the worker thread before it is launched, to ensure only the necessary data are passed to the thread, thus gives the program the rigor of the logic.

6.4 Optical Character Recognition(OCR)

As the main goal of the project is to build the MIPBR, it is important to implement language detection on the image text first and foremost, followed by text extraction. In this case, Tesseract is used to achieve this function, as its 4.0 version is in support of more than 100 languages.

6.4.1 Language Detection

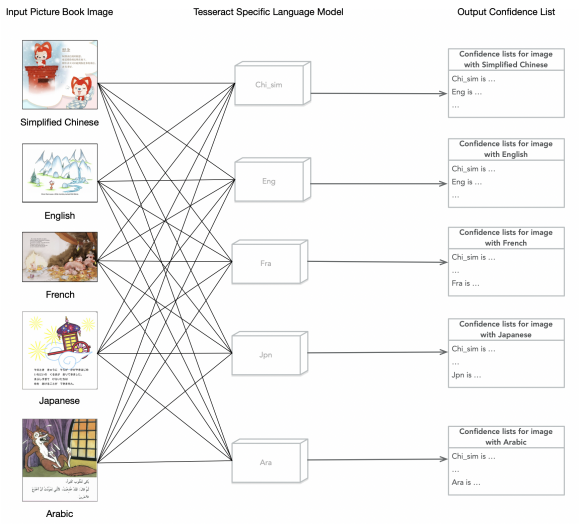


Fig. 19: Language Detection Example

Fig. 19 is an example of how language detection model works with 5 languages, where Simplified Chinese, English, French, Japanese and Arabic are chosen as input language. This is because English is heavily penetrated and affected by Norman French, and Chinese Kanji script is one of the three scripts used in Japanese. Thus, Chinese and Japanese haves similarities in scripting, as do English and French. Moreover, Arabic is a representative of right-to-left scripts. Therefore, this selection of languages contributes to the difficulty for the language detection model to distinguish one from the others.

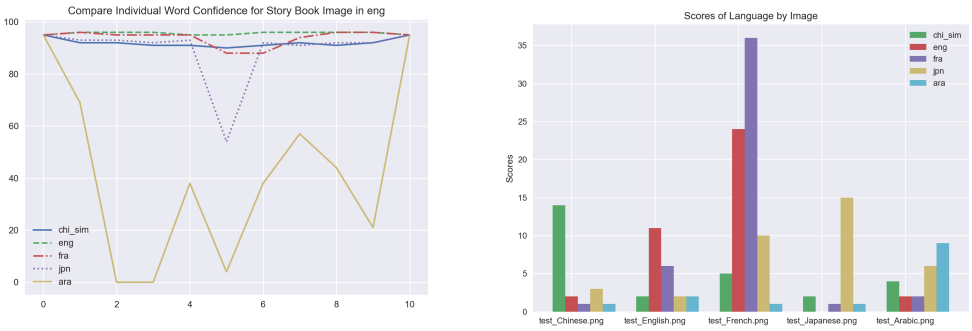
```
chi_sim confidence is : [95, 92, 92, 91, 91, 90, 91, 92, 91, 92, 95], overall confidence is 92
eng confidence is : [95, 96, 96, 96, 95, 95, 96, 96, 96, 96, 95], overall confidence is 95.63636363636364
fra confidence is : [95, 96, 95, 95, 95, 88, 88, 94, 96, 96, 95], overall confidence is 93.9090909090909
jpn confidence is : [95, 93, 93, 92, 93, 54, 92, 91, 92, 92, 95], overall confidence is 89.27272727272727
ara confidence is : [95, 69, 0, 0, 38, 4, 38, 57, 44, 21, 95], overall confidence is 41.90909090909091
```

Fig. 20: Lists of Confidence Score for Image with English Text

To detect a language which appears in the storybook image, first input those images to language specific Tesseract model to calculate lists of confidence score for that language. Then, by measuring those lists, the model could answer which is the identified language for text in the image. However, language detect model could returns lists with different lengths for different languages, and the mean of those lists could be quite close to each other. For example, as shown in Fig. 20, the mean confidence score for English Tesseract model performing its work on English is approx. 95.6, but on French it also has a relatively high score, which is approx. 93.9. Therefore, the variance between mean confidence score of these two languages is only 1.7, if the detected language is chosen based on the mean confidence score, it is possible that the model misclassifies English words as French at some point.

As is shown in Fig. 21, it is clear that the confidence scores of three different languages are considerably close to each other for the English storybook page. In contrast, using language

score, the correct language in its corresponding storybook page is dominating. Thus, the language score is proven to have better performance of the language detect model over confidence score when distinguishing the language of the storybook.



(a) Confidence Score

(b) Language Score

Fig. 21: Comparison of Different Scoring Method

When performing unit testing on the language detection function, it was found that the exaggerated text shape on the storybook cover would undermine *Tesseract* from working properly. Objectively speaking, the storybook covers are always the image where language detection takes place. To address the issue, the project implements a mechanism as insurance to load the next few pages when the detection on the storybook cover is abnormal and correct the result via majority vote.

6.4.2 Text Extraction

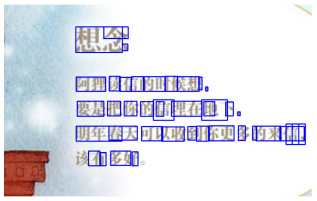
Due to the similarity in task nature, Tesseract is also used for text extraction. Unlike in language detection, where the risk of misjudgement could be circumvented through working on an alternative image, text extraction restricts work to be done on the specified image and requires a reasonable accuracy. Therefore, a considerable work has been carried out in image pre-processing to create enabling image condition for Tesseract.
Fig. 22 visualises text identified from a storybook page in Simplified Chinese, where Tesseract is doing a great job as it identifies most of the text in the image. However,two cartoon characters are identified as text, and two words are not identified, which is insufficient as a product. Hence, the Tesseract model needs to be more accurate to ignore all background cartoons and extract all foreground text. As one form of image pre-processing, image cropping could effectively mitigate background noise and provide a similar effect as zoom-in, which exposes unidentified text to Tesseract.

Notwithstanding the above mentioned image cropping, the subsequent unit testing reveals that its effect is limited if the text colour is not in strong contrast with the background colour, thus lose weights for an accurate text extraction. As is shown in Fig. 23, different thresholding methods are performed on the same grayscale image for evaluation, it is clear that global thresholding with a fixed value in Fig. 23(c) mistakenly removes text from the image, whereas adaptive Mean and Gaussian thresholding in Fig. 23(d)(e) simply retain object and text outlines while converting pixels to the same colour, the background noise still exists. Therefore, it is self-evident that Ostu thresholding in Fig. 23(f) outperforms other techniques by forming a stark contrast between foreground text and background objects.

6.5 Automatic Speech Recognition(ASR)

Fig. 24 is an overview of the speech recognition system, where three levels of models exist between the audio file and the recognised text, those models are also considered to be the

(a) Original Story book page (b) Text identified by Tesseract from image (c) After removing background cartoons



(d) Text identified in the cropped image

Fig. 22: How Tesseract pre-trained model works on different occasions

crucial components of the ASR system.

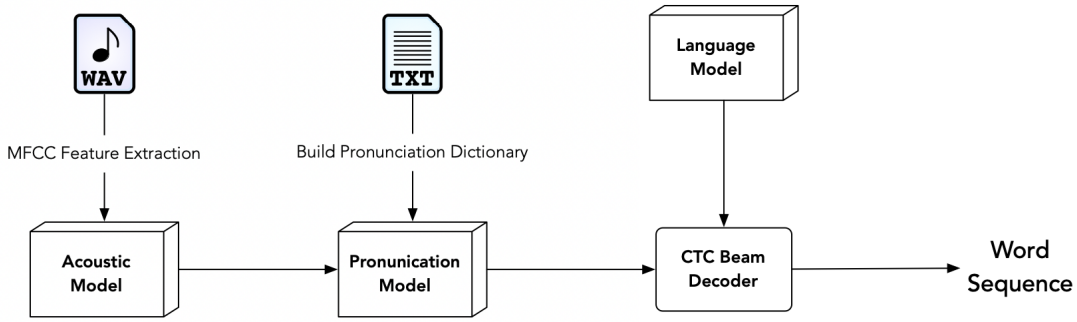


Fig. 24: Architecture of ASR system

The original voice data is loaded in a time domain format, which cannot be fed directly into a neural network. Instead, all the speech waveforms in a batch must fit a given standard length, either by padding or truncating. Namely, samples that are shorter than the longest item in the batch need to be padded with 0. One might alternatively truncate long samples before padding short ones. After that, MFCC feature extraction is performed on the inputs before the BiLSTM acoustic model is created to establish statistical representations from these samples, which is encoding in short. The parameters for training the BiLSTM model is specified in Table. 2.

The next step is to establish a map between the orthographic representation of word and the statistical representation of sound. This is done by the pronunciation model, also known as pronunciation lexicon, which maps each segment of feature vector to the corresponding text using the pronunciation dictionary built with the transcript for the training audios. Hence, a sequence of word is derived from the feature vector.

[1]The number of context samples to keep for any time domain

(a) Sample Image (b) Grayscale Image (c) Global Thresholding (V=127)

(d) Adaptive Mean Thresholding (e) Adaptive Gaussian Thresholding (f) Ostu Thresholding

Fig. 23: Different Thresholding methods

Layers	FC + FC + FC + BiLSTM + FC + FC
Activation Function	Clipped Rectified Linear Unit (ReLU)
Loss Function	CTC Loss Function
Optimizer	Adam
Learning Rate	0.001
Dropout Rate	0.95
MFCC Coefficient	26
Batch Size	8
No. Epoch	100
No. Context[1]	8

Table 2: Parameters for training BiLSTM Model

However, it is quite often that the pronunciation model could make linguistic mistakes, e.g. large duplication of words in a sentence, or ambiguate words with similar phonemes. The language model aims to tackle this issue. A context dependent language model transforms a stream of word probabilities into a stream of sentence probabilities, and pass the sentence with the highest probability to the CTC beam decoder. The output of the decoder is the recognised text.

6.6 Visual Question Answering(VQA)

6.6.1 Object Detection

The object detection model used in the project referenced the implementation of the current state-of-the-art YOLO V5 developed by Glenn et al.[28] The architecture of the object detection model is specified in Fig. 25, whereas the parameters for training the model is stated in Table. 5.

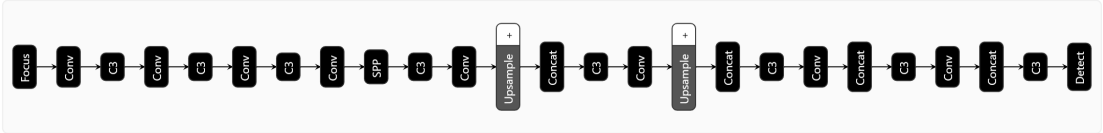


Fig. 25: Architecture of YOLO V5

Activation Function	Leaky ReLU + Sigmoid
Loss Function	Binary Cross-Entropy with Logits Loss
Optimizer	SGD
Batch Size	16
No. Epoch	500
Weights	yolov5s.pt

Table 3: Parameters for training BiLSTM Model

Due to the shortage of the training data, data augmentation is performed on the training dataset to create new training samples, thus increases the data volume and prevents the model from overfitting. The specific augmentation algorithm used in the project is the mosaic augmentation, it works by first randomly select four images from the training dataset, then performs arbitrary actions on the individual image, e.g. flip, rotate or crop, and combines these images into one with certain ratio.

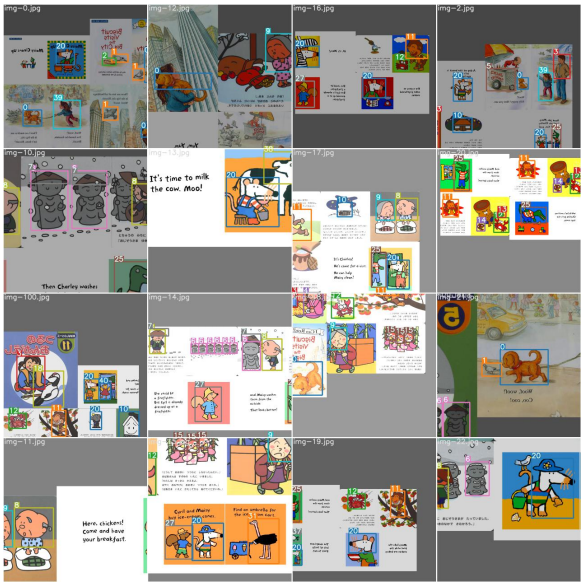


Fig. 26: Training Batch

Fig. 26 presents a batch of samples selected for one training iteration, which contains 16 samples as specified by the batch size, whereas samples are generated by the aforementioned mosaic data augmentation. The training process involves a pre-defined weights, and tunes the weights according to the custom data for 500 epochs. After that, the model exports the tuned weights that are used for cartoon image object detection for MIPBR.

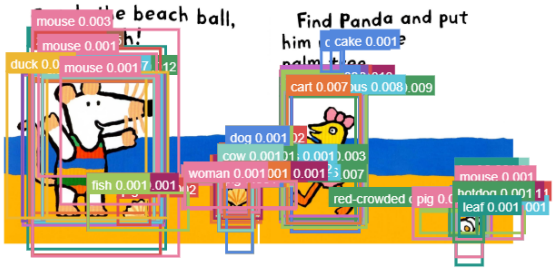


Fig. 27: Image Object Detection

As indicated in Fig. 27, when detecting objects from an image, YOLO calculates probabilities on each possible object for the classes defined in the YAML file that the weights was trained on, and labels the object by the class with the highest probability. The class with the highest probability is chosen as the label for object detected.

6.6.2 Speech Generation

As mentioned earlier, the project invokes Google Translate text-to-speech (gTTS) API for generating speech based on the given text and language.By passing these parameters to the API function, gTTS sends the text to the gTTS server and returns the corresponding speech waveform, which is then saved in a `.mp3` file locally. After the file is created, the project invoke the *playsound* function to play the `.mp3` file.

7 Evaluation

In this section, we first mentioned the performance metrics for evaluating the deep learning model performance, and using these metrics to evaluate the models, the result is expressed as graphs and tables.

7.1 Performance Metrics

7.1.1 Word Error Rate (WER)

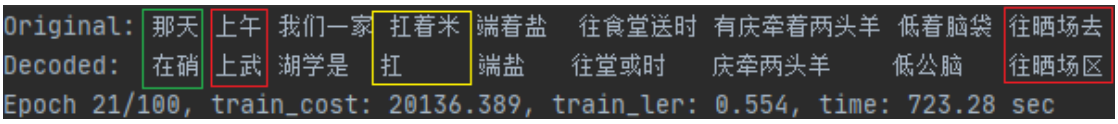


Fig. 28: Original text VS. Decoded transcript

The concept of WER is introduced to measure the performance of the speech recognition model. As can be seen from figure 7, there are three types of mis-translation when recognising a speech: substitution, insertion and deletion. WER is calculated based on those mis-translations, as follows:

$$\textit{Word Error Rate} = \frac{\textit{Substitutions} + \textit{Insertions} + \textit{Deletions}}{\textit{Number of Words Spoken}} \tag{7.1}$$

Substitution occurs when a word gets replaced, it happens mostly because of the model decodes MFCC feature vectors into the homophones of the target word, e.g. words inside the red boxes. Insertion occurs when a word from the decoded transcript does not exist in the original text, e.g. words inside the green box. Deletion occurs when a word in the original text is out of the decoded transcript, e.g. words inside the yellow box.

7.1.2 Intersection over Union (IoU)

The concept of IoU is introduced to measure the performance of the image object detection model. As its name suggests, IoU measures the area of overlapping between two bounding boxes over their area of union. In another word, it measures the correctness of a given bounding box. Commonly, a prediction could be seem as accurate if its IoU is greater than 0.5, thus 0.5 is usually used as the threshold in classifying when the prediction is a true positive or a false positive.

$$IoU = \frac{Area\ of\ Overlap}{Area\ of\ Union} \tag{7.2}$$

7.1.3 Means of Average Precision(mAP)

As mentioned earlier, YOLO calculates the probabilities of all the labels on the same object, and selects the label with the highest probability. However, even if the label has highest probability, it should be classified as false positive if the value is less than a certain threshold, eg 0.4. In this context, mAP is another performance metrics to validate whether or not label with the highest probability is appropriate for the object.

This is achieved by calculating the mean value of the area under the Precision-Recall Curve (PR Curve), if the result is less then the threshold, it is false positive. The computation for mAP is expressed as follows:

$$mAP = \frac{1}{|classes|} \sum_{c \in classes} \frac{TP(c)}{TP(c) + FP(c)} \tag{7.3}$$

7.2 Testing

Considering the main functions are empowered by the deep learning models, a great extent of functional testing is replaced by performance evaluation on the underlying models.

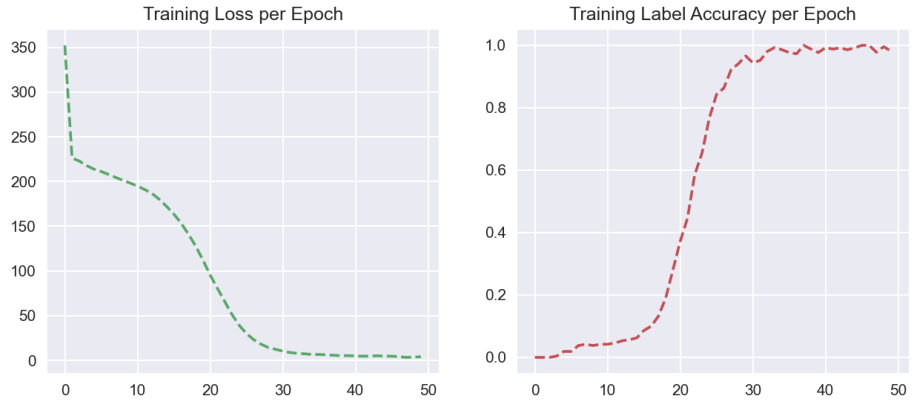


Fig. 29: BiLSTM Model Performance

Due to the time constraint and the hardware limitation, the BiLSTM model is trained on a subsample of 2000 training audios. As can be seen from Fig. 29, after 10 hours training, the training loss in the last epoch is approaching 0, with a training accuracy approaching 1. Certainly the speech recognition model has an excellent performance, but with only data size 2000, such performance might be due to the reason of over-fitting. Therefore, the model will be further trained on the entire dataset to make the model more generalisable to unseen data.

Fig. 31 visualises the text identified on the images by the text extraction model, it is worth mention that Tesseract cannot detect anything on the raw image. In contrast, all the letters

are recognised on the same image after performing Ostu thresholding, which strongly indicates the effectiveness of the proposed image pre-processing for text extraction.

Fig. 30: Text Extraction after Ostu Thresholding

Table 5 shows the result of text extraction on the training dataset for training the object detection model, where the proposed image prepossessing achieves a decent average WER over three languages.

	Raw Image	Ostu Thresholding
WER	0.42	0.25

Table 4: Text Extraction Result

As can be seen from Fig 31, dramatic fluctuation take place during the training for YOLO V5, especially in its early stage. This is because the training data is composed of storybook pages from three different culture backgrounds with distinctive art style, where the volume of images from the Japanese storybooks is greater than the others. In this case, YOLO is more familiar with this particular style for an initial period, but the fluctuation is mitigated over time with the increasing occurrence of images in others styles. This circumstance would also be alleviated when more training data are gathered.

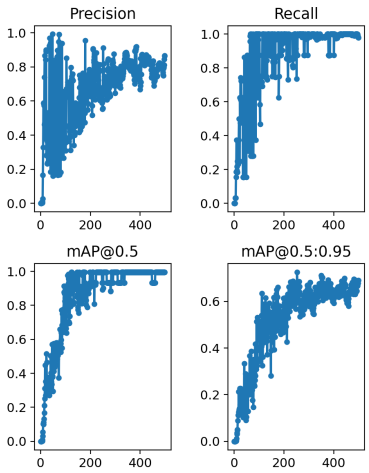


Fig. 31: YOLO V5 Performance

Table 5 states the YOLO V5 performance on the validation dataset after 500 epochs, which shows a promising performance of YOLO V5. Nevertheless, the training data only has 42 object classes, which means the performance of model would be worsen encountering unseen data.

Precision	Recall	mAP@0.5[1]	mAP@0.5:0.95[2]
0.866	0.981	0.996	0.691

Table 5: YOLO V5 Result

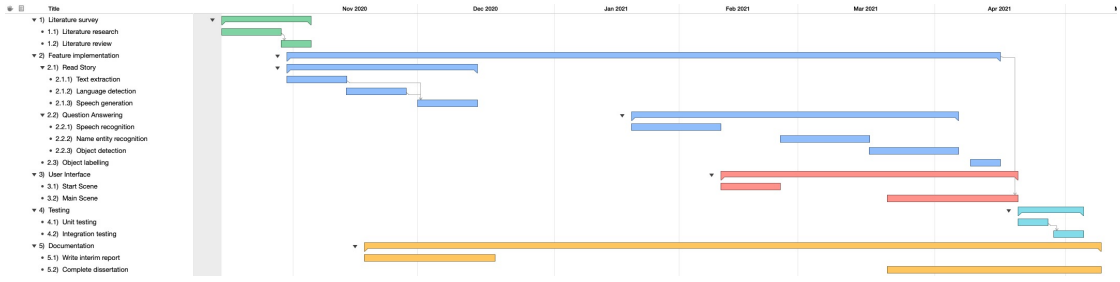
8 Summary & Reflection

8.1 Project Management

This project follows the agile scrum development methodology, where works are separated into sprints to achieve the established function or goal. The granularity for durations for each sprint is two to four weeks, with all the sprints connecting together that form the project timeline.

Fig. 32 contains two Gantt charts to visualise how the project schedule changed over time, where (a) refers to the preliminary arrangement from the project proposal, and (b) depicts how the project is managed in practice. For clarity, sprints that perform the same type of task are assigned with the same colour.

(a) Initial Project Gantt Chart



(b) Revised Project Gantt Chart

Fig. 32: Project Gantt Chart

In the autumn semester, the project was implemented smoothly according to the initial Gantt chart, followed by me suspending the project temporarily to prepare for my exams. In the spring semester, I made a certain degree of adjustments to the project arrangement.

[1] mAP with an IoU threshold of 0.5
[2] means average mAP over different IoU thresholds, from 0.5 to 0.95, step 0.05

In the preliminary plan, the project is split into four stages: literature survey, feature development, testing and report writing, whereas each stage has clear dependencies on the previous one. However, feature development, in reality, lasts throughout the entire project, even when the UI implementation has been split as an independent stage, whereas the cause is also the UI implementation. That is, to build a modern UI, the project changes the library from *tkliner* to *Pyside2*. In this case, I have to invest more time in getting familiar with QML and its signal-slot mechanism. Meanwhile, UI implementation also forces me to design the architecture for connecting the different components immediately.

After careful consideration, I decided to shorten the testing stage and adjust it to take place after the feature development. This is due to the fact that implementation of UI also requires itself to be tested at the same time, otherwise the screen does not appear properly, which also results in the relatively longer duration for UI tasks. As for the newly added labeling tool, it was assigned to the one week set-aside time retained for any undecided task. Consequently, the risk of late submission has been successfully circumvented through the aforementioned arrangement.

8.2 Contribution & Reflection

So far, the project has shown great potential in the ideas proposed within the proposal. The progress achieved in the project has already shown that the aim and objectives stated in the proposal are realistic, and has been further optimised. Initially, the project considered implementing voice cloning to avoid monotonous and emotionless robotic voices that would scare the children. However, the *gTTS* library has been proven to be a better option, which generates realistic sound as if a real human were speaking it, which makes voice cloning pointless.

Thorough literature research and review was done on the relevant academic contents, such as image processing and natural language processing. During that time, I read more than hundreds of papers and countless online researches, to find the current state-of-the-art techniques that are feasible to be deployed in the project, including BiLSTM in speech recognition, YOLO V5 in real-time object detection, and BERT in name entity recognition.

Even though the project strongly focuses on exploring the AI scenarios in the children's storybook reading, the quality of the delivered prototype implementation is not compromised. I believe that a good machine learning engineer should also be a good software engineer. Thus, in the prototype implementation, not only the deep learning models are trained and deployed, but also a modern UI that integrates a pdf viewer and an image labeling tool is implemented. To achieve this, I invested more efforts in designing the system architecture, and leveraged techniques like signal-slot mechanism to establish communication between object across Python and QML, as well as multi-processing to ensure the robustness of the software.

The project actively seeks continuous innovation and self-transcendence even if there is no similar product for reference. For example, the project uses a real-world image dataset for training the cartoon object detection model initially, and testing the model on the cartoon images, and the result is not satisfying. To strive for perfection, the project collects its own dataset, and initiates the idea of constructing a built-in labeling tool to leverage the collective strengths, which achieves promising results.

Although the project progressed smoothly, it encountered many difficulties during its process. In fact, some crucial techniques are involved in the project, e.g. speech recognition, its scope and complexity are more than enough to be treated as an independent project. Moreover, additional features added to fulfil the project also cause more burdens. Therefore, due to limitations in time and capacity, the project pursues the best performance of each component under the premise of ensuring product completion.

In general, I am very pleased that the quality of work carried out in the prototype implemen-

tation. Despite the overwhelming workload, the project's essential components met my expectations. However, I also noticed the some areas in the project that are inadequate. Thus, it is conceivable that I will keep working on the project to make it better in the future.

8.3 Further Direction

Throughout the multi-lingual interactive MIPBR's implementation, I realised some defects of the product that could be improved further, along with their preliminary designs, which are stated as follows:

- Software delivery: Although the software was developed in a conda environment that enables all the dependencies required to run the program to be exported easily, the project leverages tesseract for text extraction and CUDA to enable GPU accelerating, both these tools need users to change the system variables manually. Thus, it is tricky for people unfamiliar with programming to set up the software on their devices. In order to solve this critical problem, the project would seek an alternative to the *tesseract tool*, including implementing its own text extraction model.
- Enhanced VQA: Currently, the MIPBR only answers the user's question based on the text and object detected from the page displayed, it does not answer contextual questions related to other pages. In future, a more intelligent language model could be deployed to enhance the VQA system.
- Machine translation: Although the MIPBR can read stories in English, French, Mandarin and Japanese, and recognise voice input in English and Mandarin, it does not have the capacity to translate storybook into a different language to answer user's question. The function is achievable via implementing a seq2seq model.

References

[1] Elizabeth Margulis. “Children’s Reading in the Space Age”. In: *Montana Libraries* 8-14 (1954), p. cxxx.

[2] Fathu Rahman. “The Revival of Local Fairy Tales for Children Education”. In: *Theory and Practice in Language Studies* 7.5 (May 2017), pp. 336–344. DOI: http://dx.doi.org/10.17507/tpls.0705.02.

[3] Theresa A. Roberts. “Home Storybook Reading in Primary or Second Language With Preschool Children: Evidence of Equal Effectiveness for Second-Language Vocabulary Acquisition”. In: *International Reading Association* 43 (Nov. 2011), pp. 103–130. DOI: https://doi.org/10.1598/RRQ.43.2.1.

[4] Hua Shu and Ping Li. “Language Development and Reading Acquisition in Preschool Children”. In: *Studies in Early Childhood Education*. 10th ser. (2014). DOI: 10.13861/j.cnki.sece.2014.10.001.

[5] Jack Peat. *ONLY 30% OF PARENTS READ STORIES TO THEIR CHILDREN EVERY DAY, POLL CLAIMS*. Aug. 2018. URL: https://www.independent.co.uk/life-style/health-and-families/parents-reading-children-books-uk-roald-dahl-mcdonalds-damian-hinds-a8516436.html.

[6] *iQIYI Launches iCartoonFace Challenge Together with IJCAI-PRICAI 2020*. URL: https://www.iqiyi.com/common/20200420/5805df4bf6319b64.html.

[7] Lah Lah Banana. *Luka Reading Robot for reading Chinese books to Children*. July 2020. URL: https://lahlahbanana.wordpress.com/2020/07/07/luka-reading-robot-for-reading-chinese-books/.

[8] Daniel Jurafsky and James H. Martin. *Speech and Language Processing*. PRENTICE HALL SERIES IN ARTIFICIAL INTELLIGENCE. Prentice-Hall, Inc., Sept. 1999. ISBN: 0-13-095069-6.

[9] Jonathan Hui. *Speech Recognition — Feature Extraction MFCC & PLP*. Aug. 2019. URL: https://jonathan-hui.medium.com/speech-recognition-feature-extraction-mfcc-plp-5455f5a69dd9.

[10] Alex Acero Xuedong Huang and Hsiao-Wuen Hon. *Spoken Language Processing: A Guide to Theory, Algorithm, and System Development*. PRENTICE HALL SERIES IN ARTIFICIAL INTELLIGENCE. Prentice-Hall, Inc., Jan. 2001. ISBN: 0-13-022616-5.

[11] N. Otsu. “A threshold selection method from gray level histograms”. In: *IEEE Transactions on Systems, Man, and Cybernetics* 9 (1979), pp. 62–66.

[12] Saman Sarraf. “French Word Recognition through a Quick Survey on Recurrent Neural Networks Using Long-Short Term Memory RNN-LSTM”. In: 39 (2018), p. 18.

[13] Afshine Amidi and Shervine Amidi. *Recurrent Neural Networks Cheatsheet*. URL: https://stanford.edu/~shervine/teaching/cs-230/cheatsheet-recurrent-neural-networks.

[14] Rafal Jozefowicz, Wojciech Zaremba, and Ilya Sutskever. “An Empirical Exploration of Recurrent Network Architectures”. In: *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*. ICML’15. Lille, France: JMLR.org, 2015, pp. 2342–2350.

[15] M. Schuster and K. K. Paliwal. “Bidirectional recurrent neural networks”. In: *IEEE Transactions on Signal Processing* 45.11 (1997), pp. 2673–2681. DOI: 10.1109/78.650093.

[16] Alex Graves and Navdeep Jaitly. “Towards End-To-End Speech Recognition with Recurrent Neural Networks”. In: *Proceedings of the 31st International Conference on Machine Learning*. Ed. by Eric P. Xing and Tony Jebara. Vol. 32. Proceedings of Machine Learning Research 2. Bejing, China: PMLR, 22–24 Jun 2014, pp. 1764–1772. URL: http://proceedings.mlr.press/v32/graves14.html.

[17] Haşim Sak et al. “Learning acoustic frame labeling for speech recognition with recurrent neural networks”. In: *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2015, pp. 4280–4284. DOI: 10.1109/ICASSP.2015.7178778.

[18] Harald Scheidle. *Beam Search Decoding in CTC-trained Neural Networks*. July 2018. URL: https://towardsdatascience.com/beam-search-decoding-in-ctc-trained-neural-networks-5a889a3d85a7.

[19] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. “ImageNet Classification with Deep Convolutional Neural Networks”. In: *Commun. ACM* 60.6 (May 2017), pp. 84–90. ISSN: 0001-0782. DOI: 10.1145/3065386. URL: https://doi.org/10.1145/3065386.

[20] Ali Sharif Razavian et al. “CNN Features Off-the-Shelf: An Astounding Baseline for Recognition”. In: *2014 IEEE Conference on Computer Vision and Pattern Recognition Workshops*. 2014, pp. 512–519. DOI: 10.1109/CVPRW.2014.131.

[21] Joseph Redmon et al. “You Only Look Once: Unified, Real-Time Object Detection”. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 779–788. DOI: 10.1109/CVPR.2016.91.

[22] Zhiyong Zhang Dong Wang Xuewei Zhang. *THCHS-30 : A Free Chinese Speech Corpus*. 2015. URL: http://arxiv.org/abs/1512.01882.

[23] *Python 3.7.9 documentation*. URL: https://docs.python.org/3.7/.

[24] *Introduction to TensorFlow*. URL: https://www.tensorflow.org/learn.

[25] Pierre Nicolas Durette. *gTTS documentation*. URL: https://github.com/pndurette/gTTS.

[26] *Python-tesseract is a python wrapper for Google's Tesseract-OCR*. URL: https://pypi.org/project/pytesseract/.

[27] *Introduction to OpenCV-Python Tutorials*. URL: https://docs.opencv.org/master/d0/de3/tutorial_py_intro.html.

[28] Glenn Jocher et al. *ultralytics/yolov5: v5.0 - YOLOv5-P6 1280 models, AWS, Supervise.ly and YouTube integrations*. Version v5.0. Apr. 2021. DOI: 10.5281/zenodo.4679653. URL: https://doi.org/10.5281/zenodo.4679653.