TRANSLATION OF BAYBAYIN CHARACTERS THROUGH ARTIFICIAL INTELLIGENCE

A Research Presented to
the Faculty of Senior High School
Elizabeth Seton School-South
Anabu II – D, Imus City, Cavite

In Partial Fulfillment
of the Requirements for
Capstone

FRANCIS BERT L. DOLOT
JOSE MIGUEL A. ESCALONA
LORENZO IAN A. PAJANTOY
DEAN ANDRE M. REYES
GEOFF AARON C. TOPACIO

March 2019

ABSTRACT

Name of Institution : Elizabeth Seton School-South

Address : Anabu II-D, Imus City, Cavite

Title of Research : **TRANSLATION OF BAYBAYIN CHARACTERS THROUGH ARTIFICIAL INTELLIGENCE**

Authors : Francis Bert L. Dolot
Jose Miguel A. Escalona
Lorenzo Ian A. Pajantoy
Dean Andre M. Reyes
Geoff Aaron C. Topacio

Date Started : September 2018
Date Finished : March 2019

Preserving the nation's history and culture is one of the best ways of being a patriot to your country. Baybayin, being the Philippines' original handwriting system, was overthrown by the Spanish colonizers in replacement to the Latin form of handwriting. Today however, Senate bills which entail to promote and re-teach Baybayin as a writing system is only a step away from being enacted. Creating a tool capable of detecting and interpreting Baybayin characters is only appropriate as it is inline with the Senate Bill. Apart from this, the tool will utilize Artificial Intelligence as its backbone due the technology being often used on almost everything (Bayern, 2018). The researchers chose the method of utilizing Artificial Intelligence along with computer vision instead of purely being based on computer vision only. The development revealed that it was possible to create a system that is able to detect and interpret Baybayin characters, along with this is the respondents positive view of the system regarding the concept as useful, enticing, and satisfying as a tool which promotes in turn promotes learning and using Baybayin as a writing tool.

APPROVAL SHEET

This research entitled "**Translation of Baybayin Characters through Artificial Intelligence**", prepared and submitted by Francis Bert L. Dolot, Jose Miguel A. Escalona, Lorenzo Ian A. Pajantoy, Dean Andre M. Reyes and Geoff Aaron C. Topacio in partial fulfillment of the requirements for capstone, has been examined and is recommended for acceptance and approval for oral defense.

Jeniffer E. Paje
Research Adviser

Research Review Panel
Approved by the Committee on Oral Examination with a grade of _____.

Mr. Ferdinand Alido
Member

Mr. JoJo Potenciano
Member

Dr. Roberto T. Borromeo
Chair

Accepted and approved in partial fulfilment of the requirements in Capstone

Nenet L. Ayson, LPT
JHS/SHS Assistant Principal

ACKNOWLEDGEMENTS

This research study was not possible without the help of the people who gave their support and trust in the researchers. They would like to express their greatest gratitude and appreciation to the following:

To the One above, the researchers would like to thank Him for His guidance and wisdom that He has bestowed upon us and always for keeping us safe.

To the **researcher's parents,** they humbly grateful for all of their support and trust given to them; they thank their parents for providing them the materials and finances needed for this research. For always understanding their schedule and allowing them to meet-up to finish this research and for always believing in the researchers and bringing out their best and greatest ability.

To the **members of the Filipino and Araling Panlipunan faculty** who have given their insights about the cultural and language aspects of this research that helped the researchers to expound their prior knowledge about the topic.

To the **fellow developers, software engineers and the people in the online technology community** who have shared their ideas and much extensive knowledge in the field.

To **Ms. Jeniffer Paje**, the researchers are grateful for the efforts that they exerted throughout the whole process of this research, for the perseverance and patience that they gave to the researchers and for always believing that we can finish this study.

The product of this research paper would not be possible without all of them.

The Researchers

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

INTRODUCTION

Background of the Study

The Filipinos, like any nationality, is identified and distinguished through its set of cultures. According to Oak (2018), culture is a diverse yet a cohesive piece for social control that makes up a community. It also serves as the identity of a community, separating them from other countries (Spencer, 2014). To further expound the idea of culture, Birukou et. al. (2013) stated that culture is something that is partaken and/or acquired by a group of people by means of knowledge, belief, art, law, morals, custom and others. Simply put, culture is a unique set of identities of a nation's citizens that come in forms of knowledge which can include historical knowledge and language, belief which may be supernatural in form, art that shows the creative identity of a certain nation, morals that uphold the citizens in order for them to live better and custom which include their practices of living. All of this collectively forms an identity of the nation's people.

In relation to the potential loss of a nation's culture, in 2013 CHED passed a memorandum (CHED, 2013), postponed due to a TRO in 2015 (Buan, L., 2018) but was then lifted in 2018 by the Supreme Court where it basically stated the lost of mandation of the Filipino subject or Panitikan to be part of the college curriculum since it will be strongly integrated to the K-12 system itself. Along with this is the integration of the Korean language as part of the Special Program in Foreign Language of DepEd for the JHS and SHS (CNN Philippines, 2018). With this DepEd still stated that the decision of CHED will challenge DepEd in further strengthening their existing panitikan subjects on its K-12 program (Enano, J., 2018), according to her interview from former UST Departamento of Filipino chair Roberto, he stated "while other countries show appreciation for their language and culture, we ourselves kill our own". Advocates appeal to revoke the decision of CHED and to return the Filipino and Panitikan subjects in college (Rita, J., 2018). According to her interview with Rommel Rodriguez, he said it is truly disappointing especially for the side of the teachers, students, academics, researchers, writers and cultural advocates for the Supreme Court to bring down such decision, Rodriguez also pointed out that the decision shows a lack of understanding by the Supreme Court and CHED on Filipino subjects.

Baybayin, according to Lazaro (2009), is an alpha syllabary writing system or also known as abugida. It means that a single character or symbol in baybayin corresponds to a single letter or a syllable of today's modern alphabet.

Before Spanish colonization, the single modern alphabetical letters that corresponds to a baybayin character are all the vowels while the syllabic ones contain two or three letters starting with one or two consonants and ends with a vowel. It is said that the writing system originated from the Cham or the people who inhabited the kingdom of Champa in southern Vietnam (Wade, 1993) and was changed into a form we now know as Baybayin. According to the book of Woods (2012), during the early Spanish colonial period, Baybayin was deemed to be ineffective due to the unavailability of translating words that ends in consonants which is why the Spaniards modified and added a few characters that correspond to single lettered consonants, transforming into a new writing system.

Baybayin was mostly famous during the pre-colonial Spanish era and according to Woods (2012), Baybayin hardly survived until now mainly because of three reasons. First is that the materials it was written on such as bamboo and palm leaves did not survive nor it was supposed to. Second is that the nature of writings or the way it is delivered was not intended to last in long periods of time. Lastly is because of the Spanish intrusion. The Spaniards found out that the Philippines was both oral and literate in terms of their language and writing system but lacks greatly in literacy which makes it one of the underlying reasons on why the implementation of Spain's new writing system and those that are associated with it and uses its technology such as culture became immediate.

Today, it is a fact that the Baybayin wasn't entirely lost and is being given importance. According to the Senate Bill No. 1899 of the 16th Congress and Senate Bill No. 2440 of the 16th Congress or namely as the Baybayin Act of 2013 and the National Writing System Act of 2014, it is important and a must to accentuate the continuation of the old tradition in script writing for a few remaining indigenous communities because scripts are connected to language and literature.

This law intends to give priority to the Filipino culture and to emphasize patriotism and nationalism. It also promotes the national language of the country allowing cultural and linguistic diversity. In terms of societal progression, it can now be used as a business specifically tourism. Some netizen's in social media showed their interest in Baybayin by having their tattoos written in Baybayin, showing images that contain Baybayin characters while having a nationalistic caption, trying to practice writing Baybayin on their own using their names as an example thus proving a spark of interest amongst Filipinos. Groups like Singhabi revive the writing system by teaching them to the youth ("Pagsusulat ng Baybayin, layong buhayin ng grupong Hibla Sanghabi", 2017), according to the organization's leader Leo Emmanuel Castro (Castro, n.d.) , they are teaching it the youth so that they will learn and be able to feature the way we write before the time of the Spaniards.

Artificial Intelligence is a branch of computer science that aims to bring human intelligence to machines through certain algorithms that can mimic the human mind. According to Swarup (2012), the concept of artificial intelligence was initially thought of since the time of ancient Egypt albeit in a form of a folklore. But it was in the 1940s where it can take form already with the advent of electronic computers at that time. With this, it can be inferred that humans have long aimed to improve and simplify their lives by using different technologies. Some of the well-known example of this are the simple machines which aided the early ancestors of humans in making certain physical tasks simpler to do, the 1800s saw the use of the steam engine in accelerating the process of production of different industries thus referring to it as the industrial age, and the digital age which simplifies the work of the human mind by offloading most computational tasks to an electronic machine.

Computers in general are good at executing smart tasks but is deficient in thinking up those smart ideas (Evarista, 2015). This is where artificial intelligence takes in to place, where Evarista also stated that AI aims to take computing to a higher level while also being able to think and not just execute smart ideas. Artificial Intelligence as a concept is old according to Miailhe et. al. (2017) which can be proven since it was first coined by John McCarthy back in 1956. In order to retain it as relevant and useful as possible three trends must be converged for it to stay relevant which are big data, machine learning and cloud supercomputing, Miailhe et. al. (2017).

Big data is referred as large sets of complex data that is in a structured or unstructured form which makes traditional algorithms and solutions unable to operate well, Taylor-Sakyi (2016). Since traditional approaches cannot be used for such data problems, the technique called machine learning can be used in order to solve the problem as mentioned in earlier in the three trends. Machine Learning according to Das et. al. (2015) is a subset of AI where the machine "learns" or is "trained" on its own without coding the instructions in highly detailed manner. Simon (2015) also stated that machine learning works with the combination of pattern recognition and computational learning theory, she also stated that it works by learning and making predictions or inferences based from the data it was trained from. Machine learning being a subset of Artificial Intelligence shows a machine's capability to resolve and learn things given a learning algorithm. Simon also stated that machine learning can be applied in areas where human expertise is lacking or the problem is just too hard and incomprehensible for humans alone to solve. Learning through scanning for patterns alone is not the most efficient or even the most accurate way of learning. Traditional machine learning approaches, while good, are still limited in processing data sets that are in natural form, LeCun et. al. (2015).

Deep learning solves this problem, this subset of machine learning which comprises numerous hidden layers of artificial neural networks which are the computer version of neurons of the human brain, (Vargas et. al, 2018). As a whole, the use of machine learning with deep learning in image and object classifying, object detection and optical character recognition, in form Baybayin in this case, is one of the best ways of doing it. Given with correct reference or training data, it could detect the characters with utmost accuracy without the need of explicitly coding the features that the system must look out for.

In order to use the AI in object detection, computers must be also able to "see" things, this is where Computer Vision is applied. Computer vision, according to Learned-Miller (2013), is a field of science wherein computers and other machines are endowed with vision or the ability to see, this is by extracting information from the digital images (Krishna, 2017). Since its inception, its applications became numerous which include: face detection, scene recognition, self-driving cars, optical character recognition and others which enhance and simplify the lives of humans.

Computer or machine vision, simply needs two components which are sensing and interpreting devices (Krishna, 2017). The sensing device include the camera which gathers light to its image sensor which is then converted into electrical signals that is then transferred to the interpreting device which is the computer to be analyzed for. The computer will do all of the processing and interpretation with the use of an algorithm, which is usually specialized and designed for specific purposes. Applying artificial intelligence to the algorithm part of computer vision solves the issue of the limitations of coding specialized algorithms by allowing the computer to think on its own based from the lessons it gathered from its training which can be also applied in to other applications thus making more generic in purpose.

Using an innovative approach, the researchers hope to resolve problem of further losing the Philippines' cultural heritage by aiding in promoting the concept of Baybayin through technology. Baybayin is considered to be one of the national treasures of culture since it is truly unique in the Philippines. Promoting Baybayin aims to promote to Filipinos their own culture which in turn will give of a sense of national pride.

Conceptual Framework

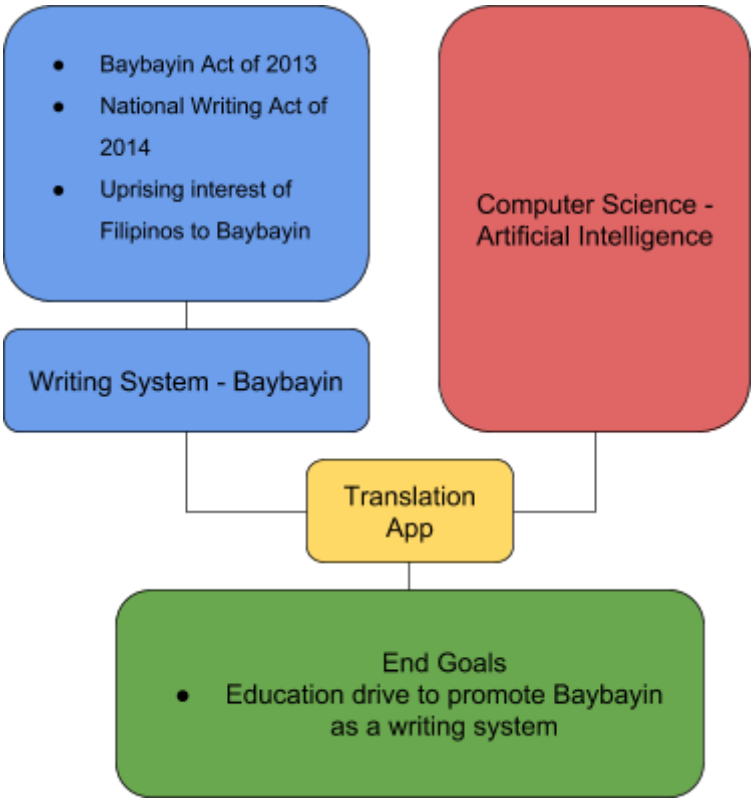


Figure 1 - Conceptual Framework

Statement of the Problem

This research aims to develop a program that will be able to detect and translate Baybayin characters into Roman/Latin syllable equivalents. The problems to be solved are as follows:

1. Is it possible to create a Baybayin Translator that utilizes artificial intelligence that is capable of interpreting and translating Baybayin characters to Roman/Latin Alphabet?
2. How accurate are the results of the translation from Baybayin characters to its equivalent traditional Filipino alphabet?
3. Is the accuracy of the prototyped system in translating images into words entice users to use the project as Baybayin translator?

Significance of the Study

This study can be beneficial to these individuals:

Archaeologists - Translation of Baybayin texts will be a lot more easier through the use of translator.

Students - Learning Baybayin can easily be done through the use of technologies such as translators.

Citizens - Due to the potential implementation of designating Baybayin as a National Writing System.

Teachers - Teaching Baybayin will be a lot more easier through the use of a translator.

Filipinos - Promotion of the use of Baybayin can be addressed as preservation of national identity and culture.

Future Researchers - Provides possible machine learning models for future researches or software to base on.Writing System, Filipino Citizens may be required to learn Baybayin.

Scope and Delimitation

The translator generated through this research is limited to the translation of Baybayin texts to Roman/Latin script and vice versa. The translator is also limited to written texts and voice translation will not be implemented anymore. The system is also limited to 17 Baybayin characters which include:

Vowel	A	E	O										
Alphasyllabirics	Ba	Da	Ga	Ha	Ka	La	Ma	Na	Pa	Ra	Sa	Ta	Wa	Ya

Table 1 - Trained Character Set

According to Lahi.PH (2018), Leo Emmanuel Castro stated that there are only 17 fundamental symbols in Baybayin which can serve as the basis for other syllabics as well.

The system is also sensitive to various factors such as but not limited to lighting, image distance or size, handwriting of the writer and the orientation of the written character.

This requires for the input to be written at an upright manner similar to how traditional letters were written in order for the system to recognize the character.

Definition of Terms

- Artificial Intelligence - a branch of Computer Science that aims for a computer to do tasks that require human intelligence.
- Characters - a letter or symbol that is either printed or written that can be read visually.
- Convolutional Neural Networks - a type of deep artificial neural networks that are used primarily for computer vision which is applied commonly in image classification, similarity clustering of images, and object recognition.
- Deep Learning - a subset of machine learning that is inspired by the structure and function of the human brain in form of artificial neural networks
- Inference - usage of the model for learning about the data
- Machine Learning or ML - an subset of artificial intelligence which provides a system the ability to learn and improve automatically through patterns and experience without explicitly programming it
- Prediction - usage of the model in predicting the outcomes from new data points
- Writing System - a system of characters or symbols that are used to represent a language

METHODOLOGY

This part of the study will project statement states the steps and procedures to be taken in order to complete the objective of the project.

Project Statement

This project aims to develop an Artificial Intelligence models that recognizes Baybayin characters through Machine Learning based Artificial Intelligence. The technology aims to help in expanding the capabilities of Filipinos to learn and regain knowledge of the former writing system called Baybayin. The AI model created in this project can be also reused for other purposes to which can be expanded with more data.

Project Deliverable

The output of this project is a detector where the system will be able to detect Baybayin characters given a camera feed. The user can use as an educational or checking tool to learn Baybayin by practicing writing Baybayin characters which the system will detect. The app will run with the AI being its main back-end. The AI model used in the app can be used in the future for further training to enhance its capability to read more variations of characters which can be applied to other uses.

Word detection, which involves multiple detections with a geometrical mapping of the characters in order to stitch the characters to a word, will be implemented as an highly optional feature of the system since its primary goal is to detect characters first.

Software and Hardware Specifications

The App

The app will be designed to run on a Windows platform for the reason that these are the most common platforms for computers.

Details of Technologies to be Used

The key technologies of this system or application will be primarily founded on Artificial Intelligence (AI) specifically Machine Learning (ML). This is for the reason that this technology, as used in a huge variety of applications, is capable of determining and predicting different types of data outputs based from an input data. TensorFlow is an open-source machine learning library or framework which will be used as the main backbone of the AI of this project. It contains all of the tools and API needed for development and training in forms of scripts which are written either in Python or C++. According to Abadi et. al. (2016), TensorFlow supports a wide range of applications which focuses on a deep neural network training, inferencing and prediction. He also stated that TensorFlow was the predecessor of Google's previous machine learning system called Distbelief and was then open-sourced in 2015 (Metz, C., 2015).

The AI training models and Object Detection method include but are not limited to Faster Regions with Convolutional Networks (Faster R-CNN), You Only Look Once (YOLO) & Single Shot Detection (SSD) which are all convolutional neural network based which can be all used and executed in TensorFlow. The neural network will come with the following feature extractors which include but are not limited to: ResNet, MobileNet and Inception architectures with MobileNet (Howard, et. al., 2017) and Inception both being a creation of Google. The training models to be selected will not be limited on what was previously mentioned but will expand further if needed.

Performance

The performance target for this set of applications revolves around its accuracy and speed, with the app being able to recognize or detect characters close to realtime.

System Requirements

The project will need a certain parameter of specification for its creation and usage. The software and hardware requirements for this application will be divided into two segments which are AI related and App related.

The AI will be trained with the use of a script created with TensorFlow in Python programming language that will read all training images for recognition. AI training is resource intensive for computers since it is done in a computationally intensive manner. This is why a high performance computer or server is recommended for this part but the researchers don't have the financial capacity so instead a standalone system will be used for the training instead.

The hardware specifications for the AI training are as follows based from the recommendation of Tensorflow and Nvidia:

Component	**Requirement**
CPU	2 cores minimum, more cores are highly recommended
RAM	2GB minimum, more RAM is highly recommended
GPU	Nvidia GPU recommended for CUDA Acceleration (Optional)
Python Version	Python version 2 or 3
Operating System	Ubuntu 16.04 or later; Windows 7 or later

Table 2.1 - Recommended System Specifications for AI Training

Based from the specifications stated above, the proponents of this project have prepared two systems for training purposes to guarantee compatibility of training and system availability (detailed version found at Appendix G).

Component	System 1	System 2
CPU Cores	2	4
CPU Threads	4	4
RAM Capacity	4GB	12GB
GPU	Intel® HD Graphics 2000 (not compatible)	GTX 1050Ti
Python version	3.6.7	3.6.7
OS	Windows 10 (x64)	Windows 10 (x64)

Table 2.2 - Specifications of Systems used in AI Training

As the training took place, it was determined that System 2 was more feasible and efficient because of its capability, capacity and performance in training with more details to be found in Appendix F which is then proven in the figure below.

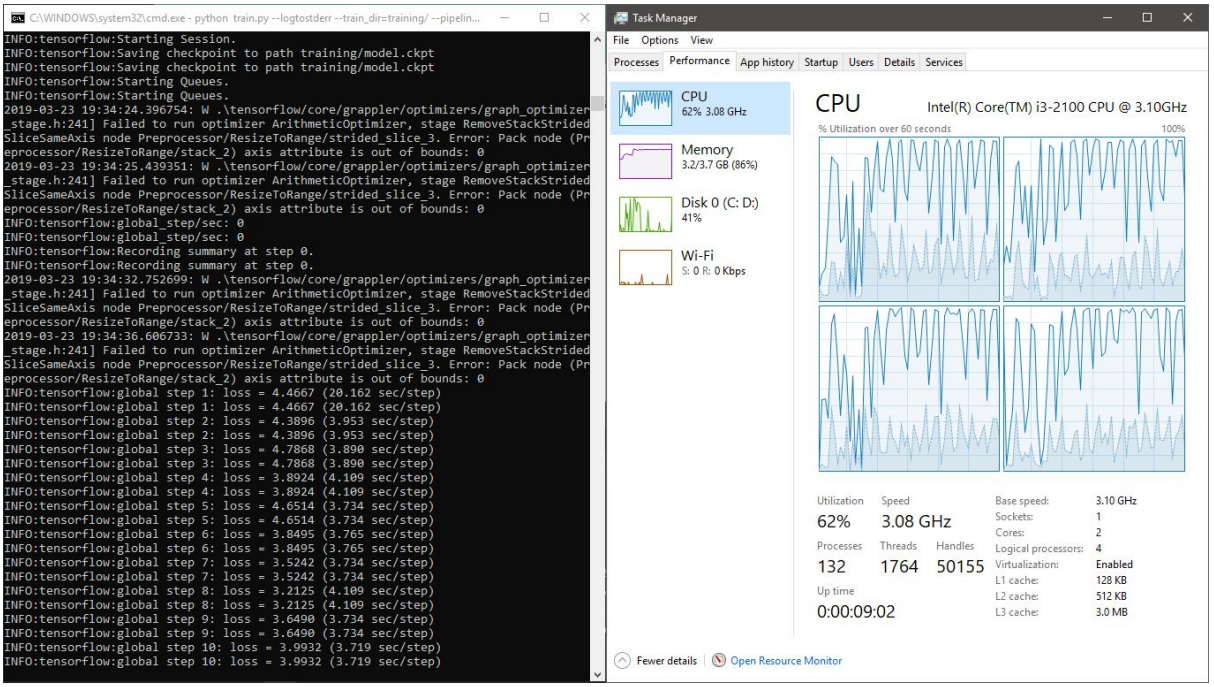


Figure 2.1- System 1 in training

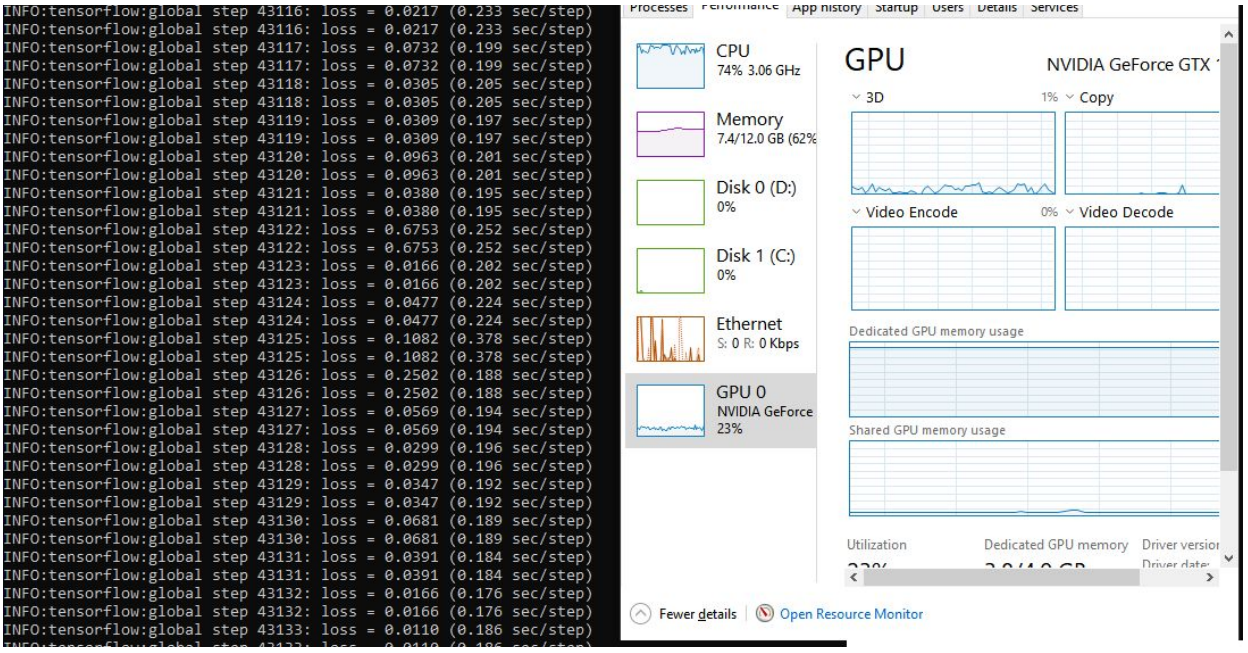


Figure 2.2 - System 2 in training

The figure shows that System 2 can train much faster at a rate of ~0.1 seconds per step in comparison, this is due to System 2 having more RAM at 12GB and having a Nvidia Graphics Card as an accelerator in comparison to the preceding system with only 4GB of RAM and being executed only by a CPU.

System Flow

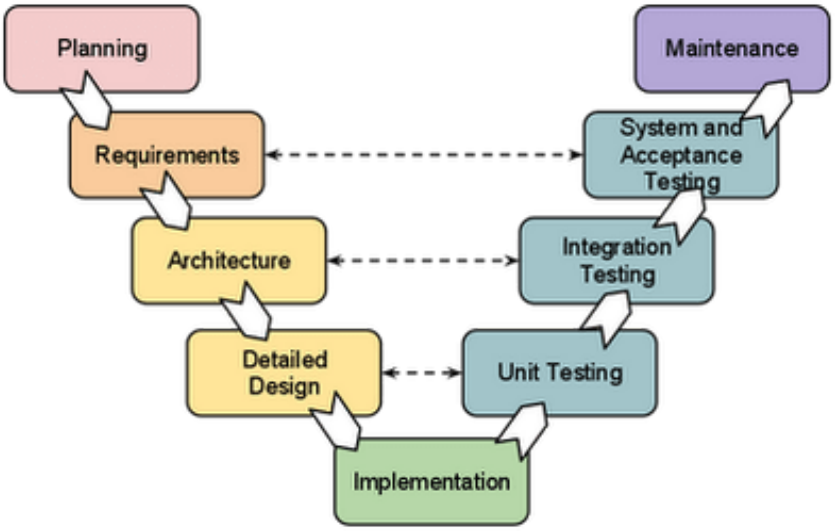


Figure 2.3 - V-Shaped Model

The development workflow that will be used for this project is the V-Shaped Model. This model is considered to be the extension of the Waterfall model used by Calasang et. al. (2017) in their software development process. According to Şaykol (2012), the V-Shaped model shows an equivalent phase of each development cycle to the testing cycles aiming to a higher success over the Waterfall model. He also stated that this is due for the reason that the testing plans are being created early on the software's life cycle. In connection, this project aims to create the software in a highly tested manner to achieve high accuracy and performance on its usage which can be done with extensive testing and optimization. Despite its advantages Şaykol also mentioned that this model suffers from flexibility issues making some adjustments difficult and expensive since each segment is dependent to one another.

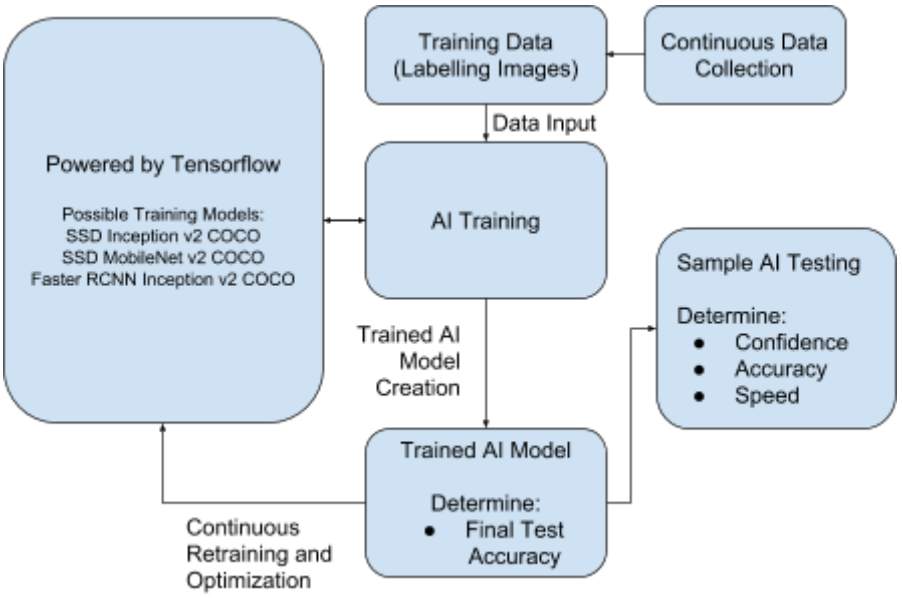


Figure 2.4 - AI Training Workflow

The AI implemented here needs training, thus a workflow is created to simplify the process of its training given two things which are the training data and the ML model this is similar to the framework shared by Mayo (2018) where there are seven steps which include:

1. Data Collection
2. Data Preparation
3. Model Selection
4. Model Training
5. Model Evaluation
6. Parameter Tuning
7. Prediction Making

The AI Model will be set at a confidence level of at least 75% to be considered confident enough, this is also to reduce excessive misdetections in the environment which it may infer as characters. For each training iteration, the AI model will be then tested using a sample image that is different or new from the trained images. This testing aims to determine the following:

- Confidence
- Accuracy
- Speed

which will dictate whether the training proceeds as intended.

Accuracy shows the correctness of the AI's interpretation. The system must achieve at least 65% successful detections for it to be considered accurate enough. The confidence shows how the system is sure of what it sees is that it thinks based from the trained model, the appropriate number that must be delivered is 75%-100% for it to be considered confident enough. The Evaluation Time meanwhile will determine how fast did the system analyze the data given, the time for evaluation must be 800ms at most.

Project Management

The project will be divided into four major segments. The first segment will be the System Design Analysis and Sub-testing to ensure that the initial system structure is correct and the AI training system functions based on the intended design. Once the system is proven to be working at a test data, the project will proceed to the next segment which is Data Collection and AI training. This segment needs data collection of different samples of Baybayin characters which will then be curated to the appropriate data clusters. The third segment will be ML model testing on a simple program to determine if the model predicts correctly based on the inputted image data and lastly, the App Development and AI optimization, the final user app will be created here which will either be on a Windows or Android platform. While the app is being developed, the ML model will still be continuously enhanced by feeding it with more data to train separate iterations to ensure development stability and model accuracy.

Requirements & Analysis

The system must be able to detect an input image for baybayin characters which will be then translated into Roman/Latin form which most people understand the most. The system has a target of no higher than 800ms evaluation time and final training accuracy of at least 85%.

System Design

The proponents designed the system to read and interpret Baybayin Characters

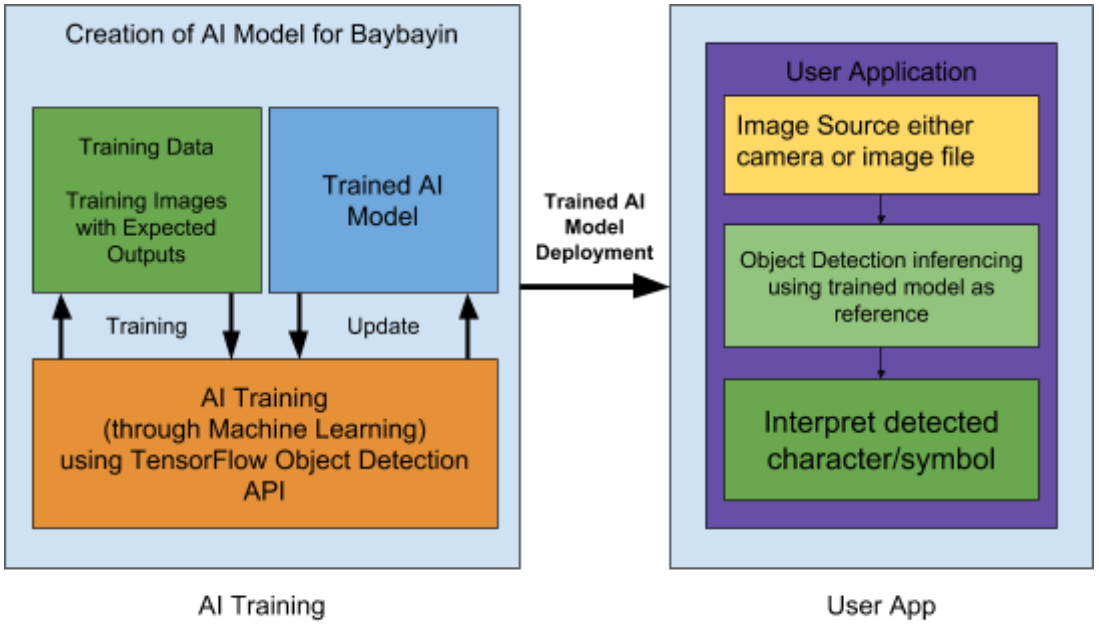


Figure 2.5 - Overall System Design

Training Model

The training model selected will be a major contributor to the system's overall performance and accuracy. The researchers have selected the Faster R-CNN Inception v2 COCO model for its exceptional performance, for training and deployment, and accuracy in comparison to other models in relation to the computer systems that the researchers are training and running from. The additional information about the training model configuration is found on Appendix M.

Implementation Plan

In order to implement and develop the project successfully, the researchers must be able to take key steps in order to achieve the desired outcome:

Step 1: Initial app development

The app will be created where it will function in its basic form, being capable of detecting characters in an image and interpreting them based from a pre-trained AI model. Implementing this will either be in a form of a desktop/laptop app or a smartphone. The development will require Microsoft Visual Studio/Java Development Kit for the desktop/laptop software and Android Studio for the smartphone. This will be done first to ensure that the app functions as it is intended before moving onto the next step of the process.

Step 2: App Testing and Data Collection

After the app has been created it will be tested based on the pre-trained AI model to determine its possible improvements and optimization.
If the system is already proven to be working on the aforementioned pre-trained AI model, data collection of sample training images of Baybayin will then commence.

The data collected, which in this case are images will be annotated and labelled to mark the parts of the image which contain the Baybayin character. The researchers written ten samples for each letter from different sources to cover the nuance of diverse handwriting by every people.

The letters were written in a one-fourth sheet of a clean bond paper either horizontally or vertically. These ten samples were then pictured on different backgrounds, lighting and a reasonable distance enough for the character to be still captured clearly. These photos were taken in highest resolution possible of the cameras which were then resized to 20% of its original size. Along with the self sourced images are carefully selected images from the internet which were resized in a similar level.

The pictures were then labeled through the use of an image annotation and labelling software. Due to the limitation of computational resources of the system used to train the model, specifically the RAM, the pictures were downscaled in to a maximum of 1024 px and a minimum of 246px.

Step 3: Pre-Training and AI testing

After the collection and processing of the training images The data will then be pre-trained with different AI training models and on different configurations. Once the desired goals have been met the training model used along with its configuration will be then used to the other parts of the training as long as it will maintain its integrity on accuracy and speed respectively. The end product of this step will be a inference graph file which the AI will use to be able to detect the characters.

Step 4: AI Training and Continuous Development

By this part of the process all forms of data including from online sources will be also included as part of the training data to ensure the system's robustness to complex inputs. Along with this, still, is the continuous development of the system which will also be constantly tested for every build.

Step 5: Integration and Validation

Nearing to the end of the entire process, the final AI model (with the highest final accuracy score) will be then selected to be integrated to the app for the final deployment which will be then tested through a systematic test called user acceptance testing (UAT) to validate that it functions as it should.

The user acceptance test were executed in the following manner. The respondents were given a brief lecture about the research and the flow of the user acceptance test. The researchers then asked the respondents to write a baybayin character that is within the scope of the model. Once the respondents were able to finish choosing and writing their desired character, they were then asked to let the system identify it. The respondents were then given a survey form to be answered whether or not the system successfully identified the character they have chosen and rate their thoughts about the concept and the system from a set of questions.

Optional Step: Continuous Training

Given that the one of the system's goal is to give an accurate detection and translation of the Baybayin characters. The AI model will be still trained and tested to further enhance its capability to discern detected images for Baybayin characters if given the time to do so.

Roles and Responsibilities

There are five proponents in this project. Each of the general tasks are to be performed by the following:

Name of the Proponent	Roles and Responsibilities
Francis Bert Dolot	Documentation and Data Gathering
Jose Miguel Escalona	AI Testing and Development, System Development
Lorenzo Ian Pajantoy	Tester and GUI Designer
Dean Andre Reyes	Documentation and Data Gathering
Geoff Aaron Topacio	System Quality Assurance and GUI Development

Table 2.3 - Roles and Responsibilities

The table shows the tasks which are distributed to each member. Each member is tasked to do the following tasks in order to finish the paper accordingly.

Cost of Investment

The cost of investment used in this project during the Development of this study is shown in the table below.

Material	**Price**
Hardware: Training PC, Webcam	Php 3,450
Hardware Upgrades: DDR3 4GB RAM (Training PC)	Php 1,300
Software Licenses: Eclipse IDE, Java SE, OpenCV, TensorFlow	Not applicable since all of the API and software used are open source and are free (as long as used for non-commercial use as stated in its licensing conditions). Though for commercial use: Java SE: $2.50/month as stated by Oracle in their revised subscription plan (Krill, 2018).

Table 2.4 - Cost of Investment

The project costed only a minimum amount since most of the major components of the system such as TensorFlow is open-source, thus free to use. The machine that will be used for training was sourced from one of the proponents of the research. The final list of software (including its versions and architecture) and hardware used in the development process will be found Appendix G.

DATA AND TESTING

This part of the study contains the Data and Testing of the research which include the data collected, methods and results.

System Performance

The performance of the prototype system will be tested on three computer systems with varying specifications.

	System 1	**System 2**	**System 3**
CPU	Intel i3-2100	Intel i7-6700HQ	AMD Ryzen 3 1200
CPU Speed	3.1Ghz	2.8	3.1
CPU Cores	2	4	4
CPU Thread	4	8	4
RAM	4GB	16GB	12GB
GPU	N/A	Nvidia GTX 960M	Nvidia GTX 1050Ti

Table 3.1 - System Specifications for Performance Testing

The performance testing was done with the following conditions:

1. The system was rebooted prior to the test
2. Latest OS updates are installed
3. All installed software versions are the same as stated in Appendix G.

Confidence Level

The confidence level of the system can be modified within the code. Initially the confidence level of the AI was set to filter 60% and below which simply means that anything that it is not confident enough will be discarded in the detections. Upon testing however, the value turned out to be erratic and continuously bringing out false detections along with the correct ones. The value was then set to 70%, this in turn reduced the erraticness of the AI in detecting and reduced the false detections to a minimum.

Public Demonstration

To determine the system's capability in detecting the predetermined set of characters, the researchers conducted a test wherein surveyees were asked to write set of characters which will be used to test the system.

Questionnaire

Following the demonstrative test, the respondents were therefore asked to answer a questionnaire to determine their thoughts with regards to the concept and the system itself.

The questionnaire intended to test the student if they have prior knowledge in our own writing system which is baybayin and to test the system ideal purpose. The type of questionnaire that the research used is in a form of Likert Scale to further see the hierarchy of the students opinion and measure's one idea about the topic.

QUESTIONNAIRE

A Questionnaire Made As An Assessment Of The Researcher-Made System Involving The Translation Of Baybayin Characters Through Artificial Intelligence S.Y 2018 – 2019

Name: _____ *(optional)* **Grade Level:** _____

Percent Value Shown: _____ **Time Before It Displayed** (In Sec.)_____

Direction: Please check the box that corresponds to your assessment on the given statements using the following scales:

5 – Strongly Agree **4** – Agree **3** – Neutral **2** – Disagree **1** – Strongly Disagree

	5	4	3	2	1
The system provides a compelling insight for the use of Baybayin.					
The system showcases the Baybayin as a distinct and interesting writing system.					
The system features a user-friendly interface that allows ease in operation thereby motivating the user to use Baybayin.					
The system gives off an overall appeal for the encouragement to use Baybayin.					
The system is recommendable to others as a Baybayin translator.					
The accuracy of the system increases the user's motivation to further use it.					
The accuracy of the system justifies the purpose of the translator.					

Figure 3.1 - Questionnaire

Respondents

The survey was done on a total of 50 respondents from 25 random highschool students and another 25 random teachers of Elizabeth Seton School - South in order to reflect the primary function of the concept and system as an educational tool. The quantity of the respondents was dictated by the Central Limit Theorem wherein it works if the sample size is more than or equal to 30.

Accuracy Testing

Determining the accuracy of the system requires the use of the success rate formula, the formula is as follows (Maloney, 2018):

$$S = \frac{A_S}{A}$$

Figure 3.2. - Success Rate Formula

S = Success Rate

A_S = Successful attempts

A = Total Attempts

This was done since the target of this test is to determine how frequent the system gets the trained characters correct. The test was done in 5 attempts per character that is included in the training for a total of 85 attempts. The testing material was gathered from random people who have been asked to copy the Baybayin characters instinctively but correctly as possible. According to Nielsen (2001), the success rate is a simple usability metric where testing can be done in a small quantities rather than huge quantities, albeit this must be done as frequent as possible. The reason for this is for the creators to be able to identify and fix a problem as soon as it is risen.

The test was done in a room with a 87 lux window lighting and 13 lux ambient overhead lighting with a distance of about 40 inches away from the camera with an variation allowance of ± 3 inches, this was done to ensure minimal environmental differences between attempts.

Results

Following the tests and surveys are the results below which are categorized according to their purpose.

Conceptual

The results of the survey reflect what people think about the concept of the system and AI in connection to Baybayin which are generally positive with scores averaging of more than 4 thus they agree on the idea of the Baybayin translator.

Questions	**Score**
1. The system provides a compelling insight for the use of Baybayin	4.52
2. The system showcases the Baybayin as a distinct and interesting writing system	4.6
3. The system features a user-friendly interface that allows ease in operation thereby motivating the user to use Baybayin	4.44
4. The system gives off an overall appeal for the encouragement to use Baybayin.	4.44
5. The system is recommendable to others as a Baybayin translator.	4.52
6. The accuracy of the system increases the user's motivation to further use it.	4.52
7. The accuracy of the system justifies the purpose of the translator.	4.64

Table 3.2 - Survey Results

Even though the questionnaire received a high score, it is inherent that the respondents, especially the teachers, orally suggested some possible improvements on the system which are:

1. Improved user experience by means of implementing the user app on a mobile device such as a smartphone or tablet.
2. Expanded character set of recognizable characters

Accuracy

Out of the 17 characters, 12 of the characters reached an accuracy of 80-100% which is more than the target 65% accuracy.

Letter/ Syllable	Attempts	Success Quantity	Rate	Accuracy
A	5	5	100.00%	Accurate
E/I	5	4	80.00%	Accurate
O/U	5	0	0.00%	Not Accurate
Ba	5	5	100.00%	Accurate
Da	5	4	80.00%	Accurate
Ga	5	4	80.00%	Accurate
Ha	5	5	100.00%	Accurate
Ka	5	5	100.00%	Accurate
La	5	3	60.00%	Not Accurate
Ma	5	5	100.00%	Accurate
Na	5	5	100.00%	Accurate
Pa	5	0	0.00%	Not Accurate
Ra	5	0	0.00%	Not Accurate
Sa	5	5	100.00%	Accurate
Ta	5	5	100.00%	Accurate
Wa	5	4	80.00%	Accurate
Ya	5	3	60.00%	Not Accurate

Table 3.3 - Individual Success Rates

It was determined that 5 characters were detected inaccurately, most of which are false detections. The following characters that it completely failed to detect correctly are O, Pa, and Ra. Some of the characters that are not considered to be accurate have a success rate.

Reasons for such misdetection are due to the possible noise included in the training data. These noises may include distance of the letter being too far, blurred images, improper handwriting and inconsistent angle.

Figure 3.3 - Training images with good accuracy

Figure 3.4 - Training images with bad accuracy

While the five aforementioned characters where not detected correctly or not completely detected at all during the test, it was detected on the preliminary testing during the development stage which means that it is also possible that the environment of the detection affects the detection of the characters.

As a whole, the entire system successfully detected 62 characters out of 85 characters which results to 72.94% accuracy.

Overall Total Attempts	Overall Total Successful Attempts	Success Rate
85	62	72.94%

Table 3.4 - Overall Accuracy

Performance and Speed

The system performed its predictions at 0.17s or 170ms which is close to realtime, this is in part to model selected, camera resolution and the computer's specifications which accelerated by its GPU. While the detection part is fast, it can be determined that the starting loading speed of the AI is quite slow since the neural network of the AI has to be prepared first before the system can be used, currently there are no possible fixes for this with the current state of technology when the research was conducted.

SUMMARY, CONCLUSIONS, AND RECOMMENDATIONS

This part of the study contains the Summary, Conclusions and Recommendations of the research. The general aim of the researchers is to promote the cultural identity of the Filipinos by means of creating an Artificial Intelligence that recognizes Baybayin characters and an application that conclusively translates the characters to consonants, vowels, and alpha syllabics. The prototype system was tested on the Junior High School students and teachers of Elizabeth Seton School in Imus City, Cavite.

Summary

In summary, the system worked as it should upon deployment and demonstration given that the respondents showed great interest and response to the concept of the system and Baybayin as a writing system. Knowing that Senate bills are only a step away from enacted as law, it is only rightful to prepare for that possibility of teaching the future students with the most technologically advanced way possible.

Despite the system having some errors in detection, it is still capable of detecting reliably even in an uncontrolled detection environment and with an 12 out of 17 characters being able to be detected accurately which results to an overall accuracy of 72.94%, it can be considered as successful considering that this is a prototype system done with limited time, data and computational resources.

Conclusions

The researchers concluded that the prototype system functioned well in accordance with the study's objectives and generally met the users' overall standards. It is proven to be possible to create an AI employed system that detects, interprets, and ultimately translates Baybayin characters to Roman/Latin Alphabets.

In terms of accuracy, the system showed a detection accuracy of 100% to the users' written Baybayin characters corresponding to letters or syllables such as A, Ba, Ha, Ka, Ma, Na, Sa, and Ta. This shows that the system is capable of detecting and translating those 8 characters with no error.

For Baybayin characters corresponding to letters or syllables such as E, I, Da, Ga, and Wa, the accuracy of detection is 80%. This means that the detection for those characters holds minimal error but can still be labeled as accurate.

For Baybayin characters that correspond to syllables such as La and Ya, the accuracy of detection is 60%. This means that the training data of the AI for these characters are insufficient and possibly incorrect or inconsistent thus resulting to errors in interpretation and detection.

For Baybayin characters corresponding to letters or syllables such as O, U, Pa ,and Ra, the accuracy of detection is 0% which is mainly due to the improper or "noisy" data.

This means that the training of the AI for these characters are highly insufficient and somehow incorrect due to the training data having a slight noise caused by incorrect angles, wrong handwriting and clearness/sharpness of the image which in turn results to the misinterpretation of characters or even not to be detected at all thus proving that the "cleanliness" of the training data is also essential in creating an accurate AI that must be taken into account.

Generalizing the detection, interpretation ,and translation process, 12 out of the 17 characters reached an accuracy of 80-100% which is more than enough to prove that the prototype system is accurate and plausible to do.

In terms of objectivity, as a product, the prototype system met the overall standards and satisfaction of its user base.

The researchers made a questionnaire containing a set of questions to evaluate the system's concept, performance, appeal, and accessibility with 0 being 'Strongly Disagree' and 5 as 'Strongly Agree'. According to the answers acquired in the given questionnaires, the majority of the sampled users agree or if not, strongly agree that the system provides a compelling insight for the use of Baybayin with an average score of 4.52, showcases the Baybayin as a distinct and interesting writing system with an average score of 4.6, features a user-friendly interface that motivates the user to use Baybayin with an average score of 4.44, gives off an overall appeal for the encouragement to use Baybayin with an average score of 4.44, and is recommendable to others as a Baybayin translator with an average score of 4.52, the system also motivates the user to further use it as a tool given its accuracy which is proved with a score of 4.52 and finally, the system's accuracy truly justifies its purpose as a translator with a score of 4.64.

Overall, it is proven that a Baybayin Translator that utilizes Artificial Intelligence with the capability of interpreting and translating Baybayin characters to Roman/Latin Alphabet is entirely possible. Consequently, the prototype system made by the researchers also concluded to be accurate enough despite the detection errors it has. Lastly, the concept of the product itself is proven to be generally useful, enticing, and satisfying according to the sampled users' responses.

Recommendations

Based from the conclusion above the researchers determined the following recommendations that future researchers must take in consideration if they would like to conduct a similar study.

As a concept, the researchers recommend the following as a guide for future researchers to consider in making a system of a similar concept.

1. **Make the application easier to use** by implementing in a simpler method such as executing it **through a portable device such as a smartphone or tablet**.
2. Add more support for **more characters to be recognized**.

As a prototype system, the researchers recommend the following as a guide for future researchers to consider both for another prototype or for the production version.

1. In training the AI, it is highly recommended to feed **training data (images) that are not blurred to easily detect the contours of the characters, different in font styles or handwriting to recognize various forms of the characters, varying distances or sizes, and varying in lightings for the system to also not be limited in terms of detection.** According to "Data Preparation and Feature Engineering in ML" (n.d.), the quality and quantity of the training data will be the most important component of machine learning since it learns by finding patterns in data to make predictions on new data points, thus if the initial pattern is obscured or wrong then it will predict incorrect results as well which proves the point from the conclusion that "clean" data is essential. **The researchers recommend to get samples from various respondents by making each of these respondents write each Baybayin characters at least once using their own natural handwriting.** This can solve the issue of the system strictly recognizing a limited range of handwriting variations. Also it is recommended to **train from hundreds of images if not thousand to further enhance the AI's resiliency** in detecting he characters.
2. It is also recommended to **expand the range of characters that includes all the syllabics** for the purpose of the system to be used as a translator by which it must be able to read all characters including vowels and all of its syllabics especially for the modern version of the Baybayin.
3. The training system of the AI requires a significant amount of computing power so for the hardware side, it is important to consider **using a computer or a server with more CPU cores, more RAM, and/or accelerators such as a Tensor Processing**

Units (TPU) or Graphics Processing Units (GPU) to handle the heavy computational load. With more computing power, this enables the researchers to procure more samples to be used as a reference for the artificial intelligence to train resulting in a much more accurate system whilst training in a faster rate. The researchers can also modify the training script to be more fine-tuned to the system specifications that they have ("TensorFlow Guide", n.d.).

4. For the user application, a **use of other programming languages** are also recommended, **with Python being a highly recommended language**, which is open-source and extensively supported by TensorFlow.
5. The interface design and user experience of the overall system is also a crucial aspect that needs to be considered. Attracting the users to utilize the system is part of an essential goal instilled in this research and possibly future researches that are in similarity. It is also recommended, according to some users to implement **the system or application into mobile devices for it to be more portable and simpler to use.**
6. Lastly, it is also recommended by the researchers to **embed the assets and other related files within the program or a predetermined file directory upon installation.** This is to simplify the installation process of the whole system assuming that it will be adapted, referenced, shared or used by another party.
7. Additional details about system related recommendations is found in Appendix K.

As the technology improves, eventually this system will be more portable, functional, accurate and faster than it is currently.

REFERENCES

A. Conference Paper

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., ... & Kudlur, M. (2016, November). Tensorflow: a system for large-scale machine learning. In OSDI (Vol. 16, pp. 265-283).

Şaykol, S. (2012). An Economic Analysis of Software Development Process based on Cost Models. *INTERNATIONAL CONFERENCE ON EURASIAN ECONOMIES*. Retrieved from https://www.researchgate.net/publication/281270700

Szegedy, C., Ioffe, S., Vanhoucke, V., & Alemi, A. A. (2017, February). Inception-v4, inception-resnet and the impact of residual connections on learning. In Thirty-First AAAI Conference on Artificial Intelligence.

B. Books

Spencer, S. (2014). Race and Ethnicity - Culture, Identity and Representation. London, U.K.: Routledge

Birukou A., Blanzieri E., Giorgini P., Giunchiglia F. (2013) A Formal Definition of Culture. In: Sycara K., Gelfand M., Abbe A. (eds) Models for Intercultural Collaboration and Negotiation. Advances in Group Decision and Negotiation, vol 6. Springer, Dordrecht

C. Class Notes and Lectures

Learned-Miller, E. G. (2011). Introduction to computer vision. University of Massachusetts, Amherst.

Krinsha, R. (2017). Computer Vision: Foundations and Applications [PDF Document]. Retrieved from Stanford Computer Vision Lab Website: http://vision.stanford.edu/teaching/cs131_fall1718/files/cs131-class-notes.pdf

D. Government Publication

BAYBAYIN ACT OF 2013, S. 1899, 16th Cong. (2013)

CHED. (2013). *General Education Curriculum: Holistic Understandings, Intellectual and Civic Competencies*. Manila: Office of the President

NATIONAL WRITING SYSTEM ACT OF 2014, S. 2440, 16th Cong. (2014)

E. Journal Article

Das, S., Dey, A., Pal, A. & Roy N. (2015). Applications of Artificial Intelligence in Machine Learning: Review and Prospect. *International Journal of Computer Applications*, 115(9), 31-41

Evarista, N. & Homti, N. (2015). Artificial Intelligence - Now and the Future. *International Journal of Computer Applications*, 114(18), 22-25

Miailhe, N. & Hodes, C. (2017). The Third Age of Artificial Intelligence. *Field Actions Science Reports*, 17 [Special Issue], 6-11

Simon, A., Deo, M., Venkatesan, S., & Babu, D. (2015). An Overview of Machine Learning and its Applications. International Journal of Electrical Sciences & Engineering, 1(1), 22-24.

Swarup, P. (2012). Artificial Intelligence. International Journal of Computing and Corporate Research, 2(4). Retrieved from https://www.ijccr.com/july2012/4.pdf

Wade, Geoff. (1993). On the Possible Cham Origin of the Philippine Scripts. *Journal of Southeast Asian Studies*, 24(1), 44-87

Woods, D. (2012). Baybayin revisited. PHILIPPINIANA SACRA, 47(139), 10-17.

F. Online

Alemi, A. (2016). Improving Inception and Image Classification in TensorFlow. Retrieved from https://ai.googleblog.com/2016/08/improving-inception-and-image.html

Bayern, M. (2018). How AI is spreading everywhere with the rise of smart machines. Retrieved from https://www.techrepublic.com/article/how-ai-is-spreading-everywhere-with-the-rise-of-smart-machines/

Buan, L. (2018). Supreme Court lifts TRO: Filipino not required subject in college. Retrieved from https://www.rappler.com/nation/216393-supreme-court-lifts-tro-filipino-not-required-subject-college

Castro, L. (n.d.). Leo Emmanuel Castro. Retrieved from https://www.linkedin.com/in/leo-emmanuel-castro-6508416a/

"Data Preparation and Feature Engineering in ML". (n.d.). Retrieved from https://developers.google.com/machine-learning/data-prep/

Enano, J. (2018). Filipino professors decry SC ruling on K-12. Retrieved from https://newsinfo.inquirer.net/1053117/filipino-professors-decry-sc-ruling-on-k-12

Head, K. & Mayer, T. (2009). Restrictions on trade in audiovisual services: Whom are we protecting from what?. Retrieved from https://voxeu.org/article/cultural-imports-foreign-influences-and-domestic-Traditions

Howard, A. & Zhu, M. (2017). MobileNets: Open-Source Models for Efficient On-Device Vision. Retrieved from https://ai.googleblog.com/2017/06/mobilenets-open-source-models-for.html

Krill, P. (2018). Oracle now requires a subscription to use Java SE. Retrieved from https://www.infoworld.com/article/3284164/oracle-now-requires-a-subscription-to-use-java-se.html

Lahi.PH. (2018). Mahirap ba matuto ng Baybayin? [Facebook Video Post]. Retrieved from https://www.facebook.com/Lahi.PH/videos/417297688813792/?v=417297688813792

Lazaro, D. (2009). The Fundamentals of Baybayin. Retrieved from http://bakitwhy.com/articles/fundamentals-baybayin

Maloney, L. (2018). How to Calculate Success Rate. Retrieved from https://sciencing.com/calculate-success-rate-8092890.html

Mayo, M. (2018). Frameworks for Approaching the Machine Learning Process. Retrieved from https://www.kdnuggets.com/2018/05/general-approaches-machine-learning-process.html

Metz, C. (2015). GOOGLE JUST OPEN SOURCED TENSORFLOW, ITS ARTIFICIAL INTELLIGENCE ENGINE. Retrieved from https://www.wired.com/2015/11/google-open-sources-its-artificial-intelligence-engine/

Nielsen, J. (2001). Success Rate: The Simplest Usability Metric. Retrieved from https://www.nngroup.com/articles/success-rate-the-simplest-usability-metric/

Oak, M. (2018). A Brief Overview of the Importance of Culture. Retrieved from https://historyplex.com/importance-of-culture

Pagsulat ng Baybayin, layong buhayin ng grupong Hibla Sanghabi. (2017). Retrieved from https://www.gmanetwork.com/news/video/stateofthenation/424850/pagsusulat-ng-baybayin-layong-buhayin-ng-grupong-hibla-sanghabi/video/

Public schools to teach Korean in high school. (2017). Retrieved from http://cnnphilippines.com/news/2017/06/24/Public-schools-to-teach-Korean-In-high-school.html

Rita, J. (2018). Advocates to appeal return of Filipino subjects in college. Retrieved from http://www.gmanetwork.com/news/news/nation/674611/advocates-to-appeal-return-of-filipino-subjects-in-college/story/

TensorFlow Guide. (n.d.). Retrieved from https://www.tensorflow.org/guide

G. Research Paper

Taylor-Sakyi, K. (2016). Big data: Understanding big data. arXiv preprint arXiv:1601.04602.

H. Review Article

Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., ... & Adam, H. (2017). Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*.

LeCun, Y., Bengio, Y. & Hinton, G. (2015). Deep learning. doi:10.1038/nature/14539

Vargas, R., Mosavi, A. & Ruiz, L. (2017). DEEP LEARNING: A REVIEW. Retrieved from https://www.researchgate.net/publication/318447392

APPENDIX A - TENSORFLOW SUB-DEVELOPMENT AND TESTING: TENSORFLOW IMAGE RECOGNITION TRAINING AND TESTING (CPU) via Image Classification Method

Sub-Development and Testing for Application in:
TRANSLATION OF BAYBAYIN CHARACTERS THROUGH ARTIFICIAL INTELLIGENCE

Creation Date: November 2018
Revision Date: November 2018

System Specifications (@ testing, via VM):

- OS: Lubuntu 18.10 (an Ubuntu derivative)
- VM Software: Oracle VM VirtualBox 5.2
- RAM: 2GB RAM (via VM)
- CPU: 4 Cores (via VM, with 100% max utilization limit; 3.10Ghz Host Frequency)
- GPU: Shared; 128MB VRAM
- HDD: 24GB Virtual Disk Image (VDI), with dynamic allocation

Host System Specifications:

- CPU: AMD Ryzen 3 1200 (3.10 Ghz, 4C4T)
- RAM: G.Skill 12GB RAM DDR4
- Motherboard: MSI B350M Mortar
- GPU: Nvidia GTX 1050Ti 4GB
- Storage: WD 120GB SSD + WD 1TB HDD

Revised System Specifications (November 10-12, 2018):

- OS: Ubuntu 18.04 (ubuntu-18.04.1-desktop-amd64)
- CPU: Intel i3-2100 (3.10Ghz, 2C4T)
- RAM: 2GB DDR3 1066Mhz Single-Channel, upgraded to 4GB DDR3 1066Mhz Single-Channel
- GPU: Integrated
- Storage: 500GB Hard Drive 7200rpm

Contents (not in particular order):

1. TensorFlow Installation (Linux & CPU)
2. TensorFlow Scratch Testing 1 (2nd Test: Celebrity Comparison)
3. Hardware Update
4. Linux Root Access
5. Notepaddqq Installation

Total Time of Testing: ~40+mins

Instructions (per Google Codelab Demonstration):

1) Install Lubuntu @ PC

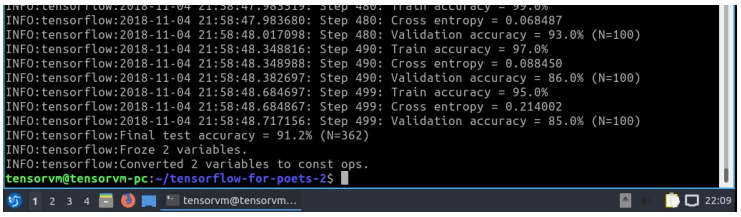
2) Follow Notepadqq Installation

1. sudo snap install --classic notepadqq

3) Install TensorFlow and others (via Ubuntu, **Python3**)

1. Source: https://www.tensorflow.org/install/pip
2. Terminal (Installation)
   - i) sudo apt update
   - ii) sudo apt install python3-dev python3-pip
   - iii) sudo pip3 install -U virtualenv # system-wide install
3. Virtualenv Install (Optional, since there is a dedicated training PC; all Terminal)
   - i) virtualenv --system-site-packages -p python3 ./venv
   - ii) source ./venv/bin/activate # sh, bash, ksh, or zsh
   - iii) pip install --upgrade pip
   - iv) pip list # show packages installed within the virtual environment
   - v) deactivate # don't exit until you're done using TensorFlow

4. Install TensorFlow pip package
   i) pip3 install --user --upgrade tensorflow # install in $HOME
   ii) python3 -c "import tensorflow as tf; tf.enable_eager_execution(); print(tf.reduce_sum(tf.random_normal([1000, 1000])))"
5. Installing TensorFlow (again, version 1.7.x from Google)
   i) Site: https://codelabs.developers.google.com/codelabs/tensorflow-for-poets/#1
   ii) Terminal: pip install –upgrade "tensorflow==1.7.*"
      (1) Warning showed up at our test whereas it recommended to use "sudo apt install python-pip"
6. Git cloning
   i) git clone https://github.com/googlecodelabs/tensorflow-for-poets-2
   ii) cd tensorflow-for-poets-2
7. Download training images
   i) Side note: curl might be needed to install so: sudo apt install curl
   ii) Terminal: curl http://download.tensorflow.org/example_images/flower_photos.tgz \
   iii) await for ">" to appear
   iv) Terminal: "| tar xz -C tf_files"
8. Confirm availability of images
   i) Terminal: ls tf_files/flower_photos

4) (Re)Training Network
1. Remember the ff. variables
2. IMAGE_SIZE = 224;
3. ARCHITECTURE="mobilenet_0.50_${IMAGE_SIZE}"
4. Terminal: python -m scripts.retrain –h
   i) This must show help
5. Terminal:
   i) python -m scripts.retrain \
   ii) --bottleneck_dir=tf_files/bottlenecks \
   iii) --how_many_training_steps=500 \
   iv) --model_dir=tf_files/models/ \
   v) --summaries_dir=tf_files/training_summaries/"${ARCHITECTURE}" \
   vi) --output_graph=tf_files/retrained_graph.pb \
   vii) --output_labels=tf_files/retrained_labels.txt \
   viii) --architecture="${ARCHITECTURE}" \
   ix) --image_dir=tf_files/flower_photos
6. IF ERROR OCCURS
   i) Module not found: numpy
      (1) Solution: pip install numpy
      (2) Solution (if python3 and pip3): pip3 install numpy
      (3) Source: https://stackoverflow.com/questions/7818811/import-error-no-module-named-numpy
   ii) No module named tensorflow
      (1) Test: pip3 show tensorflow
      (2) Solution: bet
7. 

   Last Screenshot of Terminal @ (Re)Training

5) Deploying trained model
1. **File found at tf_files/retrained_graph.pb & tf_files/retrained_labels.txt**
2. **.pb file is the trained network itself**
3. **.txt has the labels**
4. Can use Android and iOS frameworks (found at the last part of the codelabs:TensorFlow for Poets-Next Steps
   i) TFLite Android
   ii) TFLite iOS
   iii) TFMobile Android

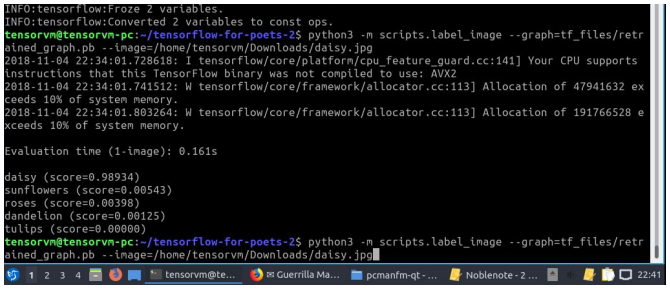
6) Testing model
1. Terminal: python -m scripts.label_image –h
   i) Should run its help module
   ii) Still use python3 instead of python only
2. Terminal:
   i) python3 –m scripts.label_image
   ii) –graph=tf_files/retrained_graph.pb
   iii) –image= *file_location*
3. Test image:

Source: https://unsplash.com/photos/xW77Twe1dvU

Image Properties:

Size: 873KB
Resolution (WxH): 4896x3264
Horizontal DPI: 72dpi
Vertical DPI: 72dpi
Bit depth: 24

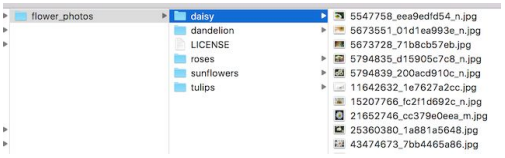
4. Test terminal:



5. Scores:
   i) Daisy: 0.98934
   ii) Sunflowers: 0.00543
   iii) Roses: 0.00398
   iv) Dandelions: 0.00125
   v) Tulips: 0.0000

7) Training on own data
   1. Found at part 8 of the codelabs: Tensorflow for Poets-Training on Your Own Categories (Optional)
   2. Since the training script works on flowers, it can learn new categories to recognize
   3. *"In theory, all you need to do is run the tool, specifying a particular set of sub-folders. Each sub-folder is named after one of your categories and contains only images from that category."* –Google
   4. Just change the –image_dir parameter to the folder of the training data

Sample Image:



   5. *"Collect as many pictures of each label as you can and try it out!"* –Google
   6. Live demo implementation example: https://youtu.be/EnFyneRScQ8?t=4m17s via ffmpeg and Android

Notes:

1) Training done with image sizes at 224px
2) MobileNet model was used which is designed for efficiency on small devices e.g. smartphones
   1. Source: https://ai.googleblog.com/2017/06/mobilenets-open-source-models-for.html
   2. Source: https://opensource.googleblog.com/2017/06/mobilenets-open-source-models-for.html
   3. Example use cases:

Tips (to prevent insanity):

1) Double check your **terminal entries**
2) Always **validate files** if error occurs
   a) **Sometimes files are just non-existent just yet**
   b) Double **check directories** for any minor issues (file structure or parameter arguments are key)
3) Don't be afraid to **use the Internet**
4) Go to the folder of all files (tensorflow) at the terminal first
5) **Just persevere!**

Future Training Development Steps:

1) Use a native machine (non-VM) to reduce virtualization overhead and to use the raw system resources.
2) Try other Tensorflow models to select the best with both accuracy and speed (in order) in mind.
3) Utilize more RAM (4GB recommended to start with).
4) Test GPU-Accelerated training via CUDA (Quite hard, instruction overhaul might be needed; https://www.tensorflow.org/install/gpu)
5) Test on more images instead of just one to validate accuracy, assuring confidence levels >85% depending on image complexity and training quantity.
6) Attain an evaluation time of <=500ms to ensure near real-time performance at live usage by testing different systems or continuously retraining training data.
7) **Find a way to create a graph (.pb and .txt) file from scratch; ie. "!retrain"**
   1. **Solution: Delete existing .pb and .txt files**

2ND TESTING: CELEBRITY COMPARISON

Date: November 6, 2018

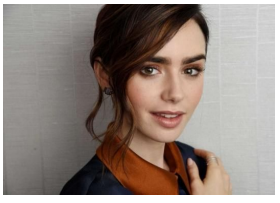
Datasets:

- Daisy Ridley = 223 images
- Lily Collins = 47 images

Initial Notes:

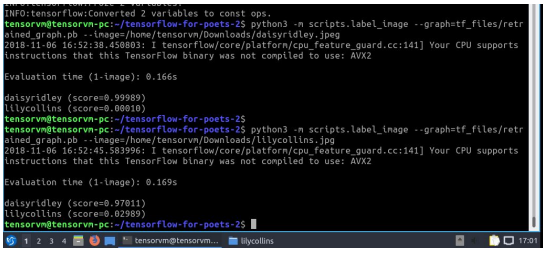
1. Training Images are gathered from the image hosting service Imgur
2. All images are searched with the following syntax:
   a. *celebname+" "+gallery+" "+imgur*
3. Downloading Websites
   a. Download in .zip form
   b. Sites
      i. imgur.rest7.com
      ii. dschep.gitub.io/imgur-album-downloader
4. All testing images are random images selected from Google Images and its respective websites with high resolutions (at least 800px on one side)

Actual Testing:

1. First Training
   a. Final accuracy = 84.8%
   b. 2 Variables
2. First Training Results
   a. Test 1.1
      i. Test Image: 
      ii. Image Specifications
         1. W = 255px
         2. H = 255px
         3. Size = 7.50KB
      iii. Label = Daisy Ridley
      iv. Scores
         1. daisyridley=1.000
         2. lilycollins=0.000
      v. Interpreted Result: Correct
   b. Test 1.2
      i. Test Image: 
      ii. Image Specifications
         1. W = 900px
         2. H = 635px
         3. Size = 284KB
      iii. Label = Lily Collins

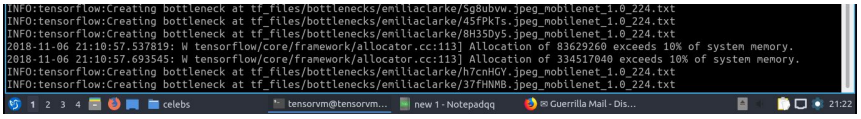
iv. Scores
    1. daisyridley=1.000
    2. lilycollins=0.000
v. Interpreted Result: Incorrect

3. First Training Errors/Issues
    a. Wrong images placed on respective folders
    b. Imbalance image quantity
    c. .pb and .txt file deleted for re-scratch
4. Second Training
    a. Fixes: removed misplaced images on each folder
    b. Final Accuracy = 76.7%
    c. 2 Variables
5. Second Training Results
    a. Same results
    b. Scores:
        i. Test 2.1
            1. Label: Daisy Ridley
            2. Same Image
            3. Score:
                a. daisyridley=0.99989
                b. lilycollins=0.00010
            4. Interpretation: Correct
        ii. Test 2.2
            1. Label: Lily Collins
            2. Same Image
            3. Score:
                a. daisyridley=0.97011
                b. lilycollins=0.02989
            4. Interpretation: Incorrect
6. Second Training Errors/Issues
    a. Imbalance number of images (223:47)
    b. Image size at terminal argument maybe incorrect
        i. Terminal: -- mobilenet_0.50_**224**
        ii. 224 being the size of test images which is not
    c. Other training models can be used as well
7. Screenshots:



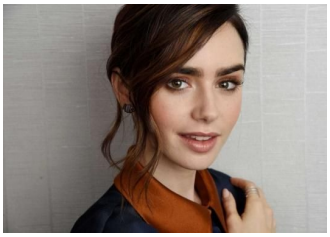
Terminal Screenshot at Testing 2.1 and 2.2

8. Second Training Evaluation Time
    a. Test 2.1 = 0.166s
    b. Test 2.2 = 1.169s
9. Optimized Retraining (Test 3.x)
    a. Notes: Image Quantity Rebalancing; New Dataset Labels (Celebrity)
    b. Dataset:
        i. Emilia Clarke = 68 images
        ii. Lily Collins = 47 images
    c. Training:
        i. Configuration
            1. Architecture = “mobilenet_1.0_224”
            2. Training Steps = 500
        ii. Final Accuracy: 90.9%
    d. Technical Notes
        i. Tensorflow Debug: Allocation of *n* exceeds 10% if system memory
        ii. Occurrences: 5
        iii. Image:

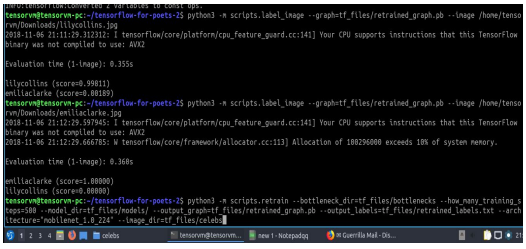


10. Optimized Testing (3.1)
    a. First Test
        i. Label: Emilia Clarke

ii. Test Image: 
iii. Image Specifications
1. W = 3500px
2. H = 2388px
3. DPI = 300dpi
4. Size = 1.85MB
iv. Interpretation = Correct
v. Scores
1. emiliaclarke=1.00000
2. lilycollins=0.00000

b. Optimized Testing (3.1)
i. Label: Lily Collins
ii. Image: 
iii. Same Specifications from Test 2.1
iv. Scores:
1. lilycollins=0.99811
2. emiliaclarke=0.00189

c. Optimized Retraining Times (3.x)
i. 3.1 = 0.355s
ii. 3.2 = 0.360s

d. Screenshots: 

11. Possible Improvements
a. Use of GPU acceleration;
b. Use Tensorboard capabilities to enhance training analysis (in form of graphs)
c. Still, use more datasets
d. Remove: “not compiled to use: AVX2”

Hardware Update: Standalone Training System (TF-01; tensorflow-01)

- System Specifications:
  - CPU: Intel i3-2100
    - Speed/Frequency: 3.10 Ghz
    - Cores: 2 cores
    - Socket Type: LGA-1155
    - Threads: 4 threads (multi-threaded)
  - RAM:
    - Standard: DDR3
    - Capacity: 2GB (to be upgraded to 6GB by adding a 4GB stick)
    - Update (11/12/18): 4GB only; RAM slot DDR3_A1 not reading memory based on testing
  - HDD:
    - Size: 3.5”
    - Capacity: 500GB
  - OS: Lubuntu 18.10 LTS (TBD for switching: Ubuntu 18.04.1 LTS)
  - Motherboard:
    - Brand: Gigabyte
    - Model: H61M-HGVS
    - Form Factor: micro-ATX (m-ATX)

- RAM slots: 2
- UEFI/BIOS Version: P1.30
- Socket Type: LGA-1155

- Image: 

Root Access on Lubuntu/Ubuntu Systems:

Video: https://www.youtube.com/watch?v=GcqP88G_vzw&

Instructions:

1. Terminal: sudo passwd root
2. Change password; “Enter new UNIX password:” should show up
3. Re-enter new root password.
4. Test using: “su –“
5. Test access

Source: https://www.wikihow.com/Become-Root-in-Linux

Recommendation: Switch to the original Ubuntu

Install htop on Ubuntu:

1) Installation:
   1. sudo apt-get update
   2. sudo apt-get install htop
2) Running: htop

System Update (11/17/2018):

OS will retain to Lubuntu 18.04 because of Python and Tensorflow issues

Running Shell Script via Terminal

1. Open terminal
2. Go to directory of the script
3. Use the “sh *filename.sh*” command on terminal

Source:

4. *Lubuntu. (n.d.). Retrieved from https://lubuntu.net/*
5. Prakash, A. (2018, June 5). 7 Best Notepad Alternatives For Linux. Retrieved from https://itsfoss.com/notepad-alternatives-for-linux/
6. For Ubuntu and most other distributions. (n.d.). Retrieved from https://notepadqq.com/wp/download/
7. Install TensorFlow with pip. (n.d.). Retrieved from https://www.tensorflow.org/install/pip
8. Tensorflow for Poets. (n.d.). Retrieved from https://codelabs.developers.google.com/codelabs/tensorflow-for-poets
9. *Train an Image Classifier with TensorFlow for Poets - Machine Learning Recipes #6* [Video file]. (2016, June 30). Retrieved from https://www.youtube.com/watch?v=cSKfRcEDGUs&
10. *How To Train an Object Detection Classifier Using TensorFlow 1.5 (GPU) on Windows 10* [Video file]. (2018, February 19). Retrieved from https://www.youtube.com/watch?v=Rgpfk6eYxJA&
11. [Digital image]. (2018, August 31). Retrieved from https://unsplash.com/photos/xW77Twe1dvU
12. On-device machine learning: TensorFlow on Android (Google Cloud Next '17)On-device machine learning: TensorFlow on Android (Google Cloud Next '17) [Video file]. (2017, March 10). Retrieved from https://youtu.be/EnFyneRScQ8?t=4m17s
13. MobileNets: Open Source Models for Efficient On-Device Vision. (2017, June 14). Retrieved from https://opensource.googleblog.com/2017/06/mobilenets-open-source-models-for.html
14. Tensorflow for Poets 2: TFLite Android. (n.d.). Retrieved from https://codelabs.developers.google.com/codelabs/tensorflow-for-poets-2-tflite/
15. How to Become Root in Linux. (n.d.). Retrieved from https://www.wikihow.com/Become-Root-in-Linux
16. *How to activate root user account in Lubuntu* [Video file]. (2016, March 18). Retrieved from https://www.youtube.com/watch?v=GcqP88G_vzw

APPENDIX B: TENSORFLOW SUB-DEVELOPMENT AND TESTING:
Comparison between Image Recognition and Object Recognition

Sub-Development and Testing for Application in:
TRANSLATION OF BAYBAYIN CHARACTERS THROUGH ARTIFICIAL INTELLIGENCE

Creation Date: November 2018
Revision Date: November 2018

Definition:

According to Zhao et. al. (2017) , object detection is capable to give valuable data or information for semantic comprehension of images and videos. Initially this system is same to image classification while having the capability to detect multiple instances of features to distinguish.

Visual Comparison:

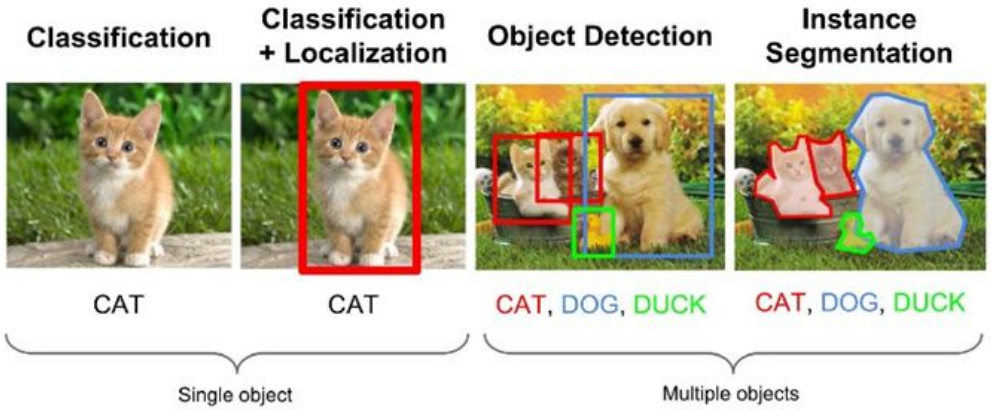


Figure 1: Image Classification vs Object Detection

From the definition earlier stated, the figure above shows the key visual differences of Image Classification and Object Detection. According to Maj (2018), object detection

answers the question “what is it and where it is?” as supposed to only asking “what is in the picture?” which is also found in the much simpler image classification. The object detection system’s task is to find and locate the all relevant objects within the image with the use of bounding boxes using object localization. In computer science, there are many ways to approach localization. According to Ibrahim et. al. (2012), the use of bounding boxes is just one of the several ways of finding the exact locations of objects in the image which include the object’s contours or by the object’s pixel.

Conclusion:

In order for the system to be able to detect multiple characters in an image environment, object detection must be used to instead of previously mentioned image classification. While both methods are fundamentally the same, the method’s capability to detect objects (which in this case are characters) at exact locations makes it more viable option than the previous method conducted.

Sources:

Ibrahim, M., Badr, A., Abdallah, M. & Eissa, I. (2012). Bounding Box Object Localization Based On Image Superpixelization. Retrieved from https://www.sciencedirect.com/science/article/pii/S1877050912007260

Maj, M. (n.d.). Object Detection and Image Classification with YOLO. Retrieved from https://www.kdnuggets.com/2018/09/object-detection-image-classification-yolo.html

Ouaknine, A. (2018). Review of Deep Learning Algorithms for Object Detection. Retrieved from: https://medium.com/comet-app/review-of-deep-learning-algorithms-for-object-detection-c1f3d437b852

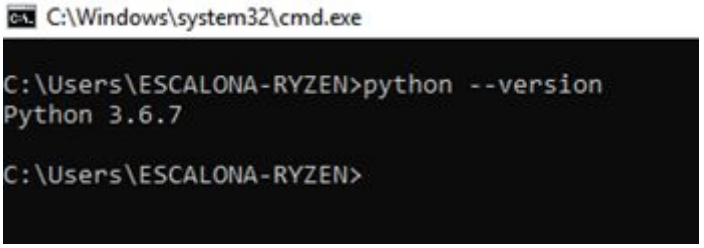
Zhao, Z. Q., Zheng, P., Xu, S. T., & Wu, X. (2018). Object detection with deep learning: A review. arXiv preprint arXiv:1807.05511

APPENDIX C: TENSORFLOW SUB-DEVELOPMENT AND TESTING:
TensorFlow Installation on Windows 10

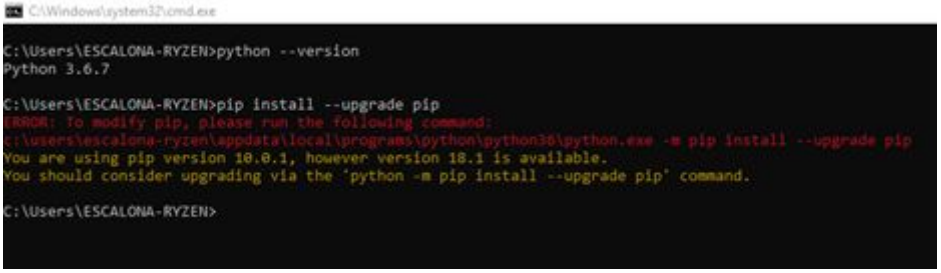
Sub-Development and Testing for Application in:
TRANSLATION OF BAYBAYIN CHARACTERS THROUGH ARTIFICIAL INTELLIGENCE

Creation Date: December 2018
Revision Date: December 2018

1. **Software needed:**
    a. Python 3.6.x
    b. Windows 10
2. **Installation:**
    a. Install Python version 3.6.x
        i. Retain installation defaults
    b. Restart system
    c. Open cmd and validate Python installation
        i. Command: *python --version*
        ii. Image:

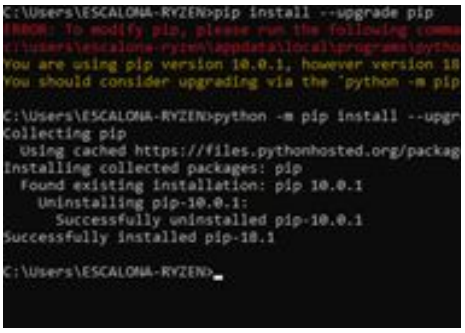


    d. Update pip
        i. Officially: *pip install --upgrade pip*
            1. Error:



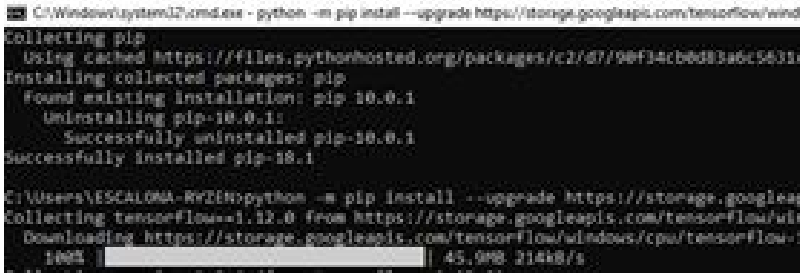
        ii. Use command: *python -m pip install --upgrade pip*

iii. After fix:



e. Download and install tensorflow: *python -m pip install --upgrade* https://storage.googleapis.com/tensorflow/windows/cpu/tensorflow-1.12.0-cp36-cp36m-win_amd64.whl

f.

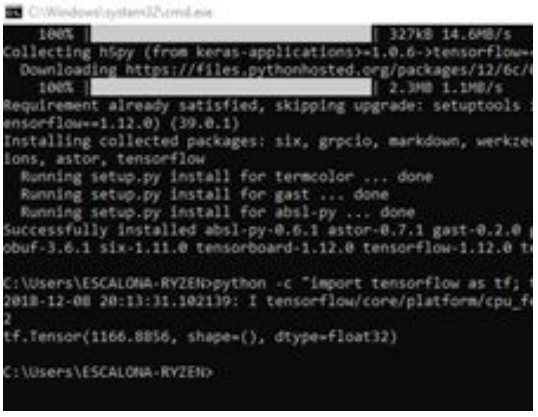


3. Validate TF install

i. Command: *python3 -c "import tensorflow as tf; tf.enable_eager_execution(); print(tf.reduce_sum(tf.random_normal([1000, 1000])))"*

ii. Image:

1.



4. Notes:

i. Use *python* instead of *python3* as instructed officially by Tensorflow because of issues with command not being found.

Sources:

Install TensorFlow with pip. (n.d.). Retrieved from https://www.tensorflow.org/install/pip

Python 3.6.7. (2018). Retrieved from https://www.python.org/downloads/release/python-367/

TensorFlow not found using pip. (2016). Retrieved from https://stackoverflow.com/questions/38896424/tensorflow-not-found-using-pip

APPENDIX D: COMPLETE JAVA ENVIRONMENT SETUP (WINDOWS)

Sub-Development and Testing for Application in:
TRANSLATION OF BAYBAYIN CHARACTERS THROUGH ARTIFICIAL INTELLIGENCE

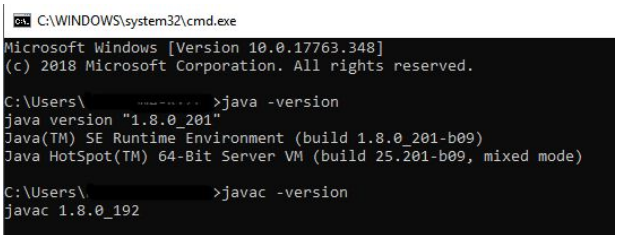
Creation Date: November 30, 2018
Revision Date: March 2019

Software needed:

1. Eclipse IDE for Java Developers (2018)
2. JDK 8 (8u202)
3. OpenCV for Java(Windows pack; 4.0.1; x64)
4. Tensorflow for Java (1.12.0)
    a. TensorFlow Jar Archive (JAR): libtensorflow.jar
    b. Java Native Interface (CPU): jni-cpu x86-64
    c. Java Native Interface (GPU): jni-gpu (optional)

Installation Procedure

1. Install JDK 8
    a. Add the bin directory to Path in System Variables in Environment Variables
2. Restart System to confirm path settings
3. Open cmd and confirm Java installation by entering: “javac -version” and “java -version”



4. Install Eclipse at a normal procedure
5. Open OpenCV and extract to a folder
6. Extract JNI to a folder, preferably at the same folder to OpenCV
7. Place libtensorflow.jar to the same folder

Including Dependencies to Project

1. Right click to project -> Properties
2. Select “Java Build Path”
3. Click “Add External JARs”
4. Adding OpenCV
    a. Locate the placement of the jar: opencv-xxx.jar (version during development is 4.0.1, thus opencv-401.jar) and click Open
    b. Expand the newly added JAR and click on Native Library Location
    c. Find the .../build/java/x64 directory of the opencv, then click OK

5. Adding TensorFlow
    a. Locate the placement of the jar: libtensorflow-x.xx.x.jar (version during development is 1.12.0, thus opencv-1.12.0.jar) and click Open
    b. Expand the newly added JAR and click on Native Library Location
    c. Find the jni folder directory of the TensorFlow, then click OKs

Pre-runtime Procedure:

1. Copy the OpenCV DLL (opencv_java401.dll) from the ...\opencv\build\java\x64 to the System32 folder in the Windows folder.
2. Copy the Tensorflow DLL (tensorflow_jni.dll) from the jni folder to the System32 folder in the Windows folder.

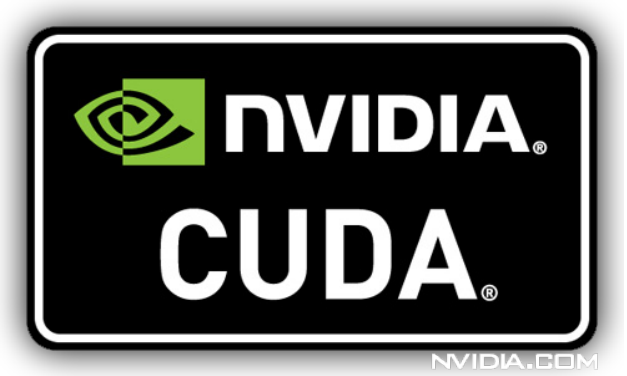
Sources:

1. Installing OpenCV for Java. (n.d.). Retrieved from https://opencv-java-tutorials.readthedocs.io/en/latest/01-installing-opencv-for-java.html
2. Install TensorFlow for Java. (n.d.). Retrieved from https://www.tensorflow.org/install/lang_java

Software Links:

1. Eclipse: https://www.eclipse.org/downloads/packages/release/2018-12/r/eclipse-ide-java-developers
2. JDK: https://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html
3. OpenCV: https://opencv.org/releases.html
4. jni (cpu): https://storage.googleapis.com/tensorflow/libtensorflow/libtensorflow_jni-cpu-windows-x86_64-1.12.0.zip
5. libtensorflow.jar: https://storage.googleapis.com/tensorflow/libtensorflow/libtensorflow-1.12.0.jar

APPENDIX E: TENSORFLOW SUB-DEVELOPMENT AND TESTING:
Adding GPU Support (CUDA) to TensorFlow (Windows)



Sub-Development and Testing for Application in:
TRANSLATION OF BAYBAYIN CHARACTERS THROUGH ARTIFICIAL INTELLIGENCE

Creation Date: March 2019
Revision Date: March 2019

Note:

This segment of development only applies if the system has a modern GPU from Nvidia (supporting drivers 410.x and above).

Software needed:

1. From pre-requisites
    a. Python 3.6.7
    b. tensorflow (CPU)
    c. Windows 10
2. Nvidia GPU Driver (410.x or higher).
3. CUDA Toolkit (9.0.176)
4. CUPTI (included in CUDA Toolkit)
5. cuDNN SDK (7.4.1.5)

Procedure:

1. Uninstall TensorFlow (CPU) from system: pip uninstall tensorflow
2. Install CUDA Toolkit
3. Follow Instructions from
    a. https://docs.nvidia.com/cuda/cuda-installation-guide-microsoft-windows/
    b. https://docs.nvidia.com/deeplearning/sdk/cudnn-install/index.html
    c. https://www.tensorflow.org/install/gpu#windows_setup
4. Reboot the system
5. Install TensorFlow (GPU) to system:
    a. Open cmd
    b. *pip install tensorflow-gpu*
6. Uninstall numpy: *pip uninstall numpy*

7. Install numpy version 1.16.2: *pip install numpy==1.16.2*

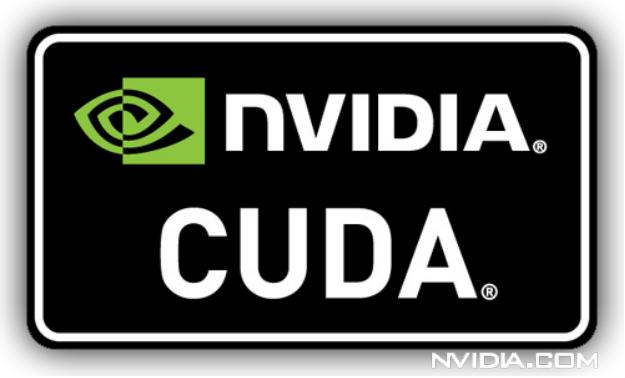
Reverting back to CPU training:

1. Open cmd
    a. pip uninstall tensorflow-gpu
    b. pip install tensorflow

Source:

GPU support. (n.d.). Retrieved from https://www.tensorflow.org/install/gpu

APPENDIX F: TENSORFLOW SUB-DEVELOPMENT AND TESTING:
TensorFlow Performance Benchmark



Sub-Development and Testing for Application in:
TRANSLATION OF BAYBAYIN CHARACTERS THROUGH ARTIFICIAL INTELLIGENCE

Creation Date: March 2019
Revision Date: March 2019

Abstract:

To determine system's viability for training a benchmark experiment was conducted to confirm its computational performance. This test was based from the test conducted by Lazorenko (2017), where he examined the performance differences between a CPU and GPU bounded training performance.

A computer has two major components that conducts computation which are the Central Processing Unit (CPU) and the Graphics Processing Unit (GPU). A CPU is microprocessor that is often referred to as the "brain" of the computer which can perform a wide range of computations while the GPU is a special type of microprocessor that is optimized to render and display graphics as well as to perform specific computational tasks (Fox, 2017). In modern times, a CPU will come in 2-8 cores per chip running in high clock speed (2-4Ghz) in normal desktops but GPUs often have thousands of cores in a single chip albeit in a lower clock speeds (500Mhz-1.2Ghz). The proponents have prepared two systems with different specifications for benchmarking.

System Specifications:

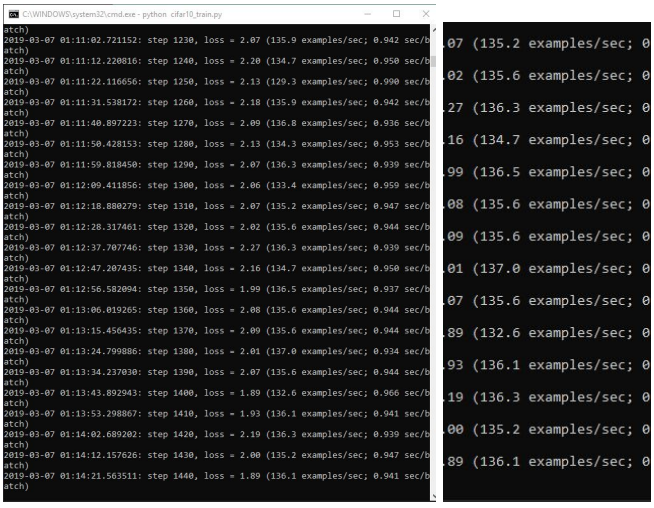
Component	System 1	System 2
CPU	i3-2100	Ryzen 3 1200
CPU Release Date	2011	2017
CPU Frequency	3100Mhz	3100Mhz
CPU Cores	2	4
CPU Thread	4	4
RAM Capacity	4GB	12GB
GPU	Intel® HD Graphics 2000	GTX 1050Ti
GPU Release Date	2011	2016
GPU Cores	N/A	768
GPU Frequency	850 MHz	1290
GPU RAM Capacity	N/A	4GB
Python version	3.6.7	3.6.7
OS	Windows 10 (x64)	Windows 10 (x64)

Benchmark Procedure:

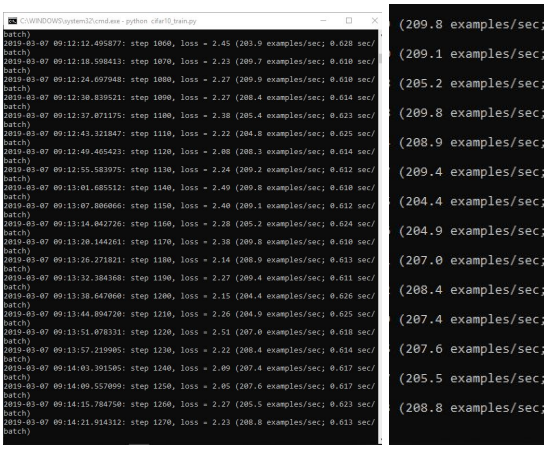
1. Download and extract: https://github.com/tensorflow/models.git
2. Go to tutorials/image/cifar10 via cmd
3. python cifar10_train.py

Results:

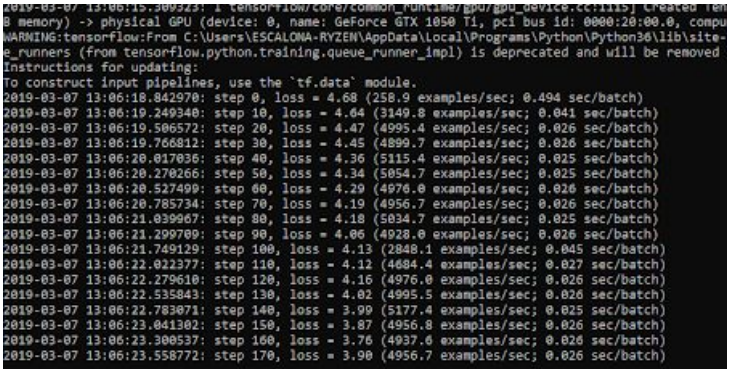
System 1 CPU: ~135 samples/sec



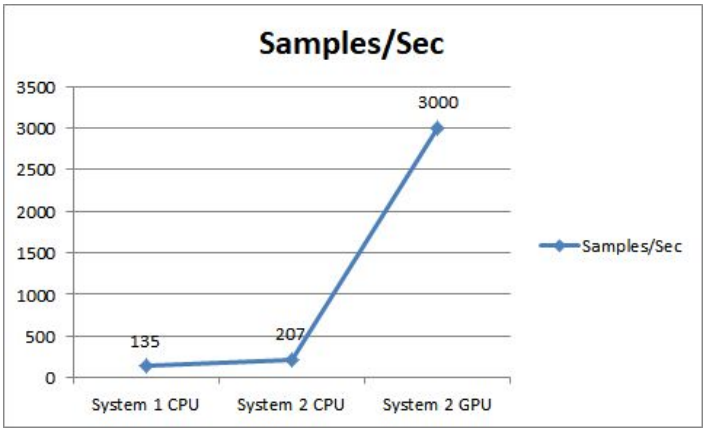
System 2 CPU: ~207 samples/sec



System 2 GPU: ~3,000 to 5,000 samples/sec



Final Result:



The experiment shows that utilizes the GPU has more performance than the other systems by almost 15 times of the CPU, thus it is highly recommended to use a GPU for training to reduce the time needed. Unfortunately, this capability is only supported by Nvidia GPUs for its CUDA support and requirement, thus it doesn't apply for AMD GPUs.

Source:

Fox, A. (2017). MTE Explains: The Difference Between a CPU and a GPU. Retrieved from https://www.maketecheasier.com/difference-between-cpu-and-gpu/

Lazorenko, A. (2017). TensorFlow performance test: CPU VS GPU. Retrieved from https://medium.com/@andriylazorenko/tensorflow-performance-test-cpu-vs-gpu-79fcd39170c

APPENDIX G: FINAL DEVELOPMENT AND RUNTIME PLATFORM

The contents of this appendix are the detailed list of software and hardware used in the development, training and runtime of this project.

Software:

Type	**Software/SDK/API/Library**	**Version**
Integrated Development Environment	Eclipse Java; Download Link: https://www.eclipse.org/downloads/packages/release/neon/3/eclipse-ide-java-developers	2018.x; can be higher than 2018
Software Development Environment	JDK SE Download Link: https://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html Note: End of Public Updates: January 2019 (Java SE. (n.d.). Retrieved from https://www.oracle.com/technetwork/java/javase/overview/index.html & Kril, P. (2018). Oracle sets date for end of Java 8 updates. Retrieved from https://www.infoworld.com/article/3269332/oracle-sets-date-for-end-of-java-8-updates.html)	1.8.0_192
Computer Vision API	OpenCV (Windows) Download Link: https://opencv.org/releases.html	4.0.1 (x64)
Machine Learning API	TensorFlow (Java) Download Link: https://www.tensorflow.org/install/lang_java	1.12.0

List of software used in Development

Type	Version	File
Java Runtime Environment	Java JRE 1.8.0_201 (x64)	Included at the installation of JDK Otherwise, download here https://www.oracle.com/technetwork/java/javase/downloads/jre8-downloads-2133155.html
OpenCV Dynamic Link Library (DLL) x64	OpenCV 4.0.1	opencv_java401.dll File size: 44.9MB
TensorFlow Dynamic Link Library (DLL) x64-CPU	TensorFlow 1.12.0	tensorflow_jni.dll File size: 58.1MB

List of software to be used in Runtime

Note: The dynamic link library files must be place in the System32 folder of Windows (C:\Windows\System32).

Type	Software/SDK/API/Library/Repositories	Version
Computer Vision API	OpenCV (Windows) Download Link: https://opencv.org/releases.html	4.0.1 (x64)
Machine Learning API	TensorFlow (Java) Download Link: https://www.tensorflow.org/install/lang_java	1.12.0
Python	Python x86-64 Download Link: https://www.python.org/downloads/release/python-367/ Packages/Modules (to be installed by pip installer): tensorflow-gpu==1.12.0 pillow lxml cython jupyer matplotlib Pandas opencv-python	3.6.7
Tensorflow Models	TensorFlow Models Repository Source: https://github.com/tensorflow/models	N/A

Model Files	Pre-trained model files Source: https://github.com/tensorflow/models/blob/master/research/object_detection/g3doc/detection_model_zoo.md	N/A
Protobuf	Protobuf compiler Source: https://github.com/protocolbuffers/protobuf/releases/tag/v3.7.0	3.7
Tensor Object Detection	TensorFlow Object Detection Repository https://github.com/EdjeElectronics/TensorFlow-Object-Detection-API-Tutorial-Train-Multiple-Objects-Windows-10	N/A

List of software used in AI Training (CPU & GPU)

Type	**Software/SDK/API/Library**	**Version**
CUDA Toolkit	CUDA Toolkit Download Link: https://developer.nvidia.com/cuda-90-download-archive File Size: 1.4GB	9.0.176 (Windows 10)
CUDA Profiling Tools Interface (CUPTI)	Included in the toolkit	N/A
cuDNN SDK	cuDNN Download Link: https://developer.nvidia.com/cudnn Note: Nvidia Developer Membership is required, though it is free.	7.4.1.5

List of additional software used in AI Training (GPU)

Type	**Software/SDK/API/Library**	**Version**	**Notes**
Image Annotator	LabelImg Source: https://github.com/tzutalin/labelImg	Windows 1.8.1	Used for annotating images Version 1.8.0 has YOLO annotating support
Image Resizing and Compressing	Caesium Image Compressor https://saerasoft.com/caesium/	1.7.0	Image file size and resolution reduction

Miscellaneous software used

All of the scripts/code below are either self-written or are sourced from the following sources/repositories:

- https://github.com/EdjeElectronics/TensorFlow-Object-Detection-API-Tutorial-Train-Multiple-Objects-Windows-10
- https://github.com/tensorflow/models
- https://stackoverflow.com/questions/11270453/how-to-remove-spaces-from-file-names-in-bulk
- https://github.com/kalaspuffar/rcnn-model-test/blob/master/src/main/java/org/ea/waldo/RunInference.java

Script/Code	**Purpose**	**Notes**
update_image_xml_csv_record.bat	Updates XML, CSV and TFrecord files of Images	Batch script
train.py	Initiates training process	Python script
generate_tfrecord.py	Converts .csv to .record files used for training	Python script; executed by update_image_xml_csv_record.bat
Object_detection_webcam.py	Runs camera with the trained model	Python script
xml_to_csv.py	Converts xml created by labelImg	Python script; executed by update_image_xml_csv_record.bat
RunInference.java	Inferencing Image	Java file

List of scripts/code used in AI Training and Runtime

Hardware:

Component	System 1	System 2 (Used for Training)
CPU	i3-2100	Ryzen 3 1200
CPU Release Date	2011	2017
CPU Frequency	3100Mhz	3100Mhz
CPU Cores	2	4
CPU Thread	4	4
RAM Capacity	4GB	12GB
GPU	Intel® HD Graphics 2000	GTX 1050Ti
GPU Release Date	2011	2016
GPU Cores	N/A	768
GPU Frequency	850 MHz	1290
GPU RAM Capacity	N/A	4GB
Python version	3.6.7	3.6.7
OS	Windows 10 (x64)	Windows 10 (x64)

Table G.6: Detailed specifications of the computers used for AI Training

Component	Developer System
CPU	Ryzen 3 1200
CPU Release Date	2017
CPU Frequency	3100Mhz
CPU Cores	4
CPU Thread	4
RAM Capacity	12GB
GPU	GTX 1050Ti
GPU Release Date	2016
GPU Cores	768

GPU Frequency	1290
GPU RAM Capacity	4GB
OS	Windows 10 (x64)

Table G.7: System Specifications used in software development

Component	**Specification**
Brand	Mikuso
Model	WC-MS-001-M
Resolution (MP)	1.3 MP
Resolution (Pixel)	640x480
Frame rate (FPS)	30

Table G.8: Camera Specifications used in Development

Type	**Configuration**
Pre-trained model	Faster R-CNN Inception v2 COCO
Image Size	196px (Minimum), 1024px (Maximum)
No. of Classes	17

Table G.9: AI Training Configuration

APPENDIX H: PROJECT STATUS AND CHANGELOG

Project Status: ONGOING
==========
Requirements(Development):
1. opencv-401.jar (https://opencv.org/releases.html)
2. libtensorflow-1.12.0.jar (https://www.tensorflow.org/install/lang_java)
3. tensorflow_jni.dll [CPU](https://www.tensorflow.org/install/lang_java; set as Native library location for

Requirements(Runtime):
1. Place OpenCV DLL file at System32 (opencv_java401.dll (x64); from https://opencv.org/releases.html)
2. Place TensorFlow dll at System32 (tensorflow_jni.dll (cpu); https://www.tensorflow.org/install/lang_java)
3. Baybayin folder (must be located in C:\, contains all runtime assets)
==========
Runtime Conditions:
1. No application using the camera must run alongside this software.
==========
Software Limitations, Technical Design & Recommendations:

Limitations:
1. Camera Resolution limited to 640x480 (1.3MP) due to GUI design constraints.
2. GUI not resizable, though a resolution based window resizing is prepared (but not implemented) in the code.
3. Requires user to manually put (if no installer is made) the DLL to System32 folder of Windows.
4. Values are in a separate class than in an XML file.
5. Limited to .jpg format of files

Design and Architecture:
1. Runs only on x64 Architecture of OS
2. Window size (width and height):
    1080p<= : 1200, 1000 to 1300,1050
    768p<= 1080p : 800,800 to 900,850

Recommendations (Software, Hardware and AI):
    Software:
        1. Enhance GUI layout
        2. Enhance GUI resizing capability
        3. Add output scaling for both Image and Camera sources (Viewport.java)
        4. Enhance webcam resolution support
        5. Add dynamic color selection in contrast from the image or camera source for bound box and text
        6. Improve system efficiency
        7. Embed assets and other related files within JAR
        8. Use other recommended languages supported by TensorFlow (Link:
           https://www.tensorflow.org/api_docs)
    Hardware:
        1. Run on powerful hardware/system
        2. Train on a system with higher specifications
            a. Possible solutions:
                i. Use of a server with more CPU cores, more RAM and/or better GPU
                ii. Use of a TPU (Tensor Processing Unit) as a training accelerator
                iii. Use of a cloud service for training for increased performance and reliability
            b. Reason: Due to RAM (CPU) and VRAM (GPU) usage during training resulting to slow
training
                or eventual crashing
    AI/Baybayin:
        1. Ensure that the data used was academically accepted set of Baybayin Symbols
        2. Include more Baybayin Symbols (Modern Baybayin)
        3. Use more datasets with the following parameters
            a. Different lighting
            b. Different (but near) font styles
            c. Note: Characters must be strictly upright since it is a reading characte

==========
To do:
- Modify graphic (boxes)

==========
Bugs:
Official Project
B001 = 10:04PM (2/18/19) - Exported runnable JAR doesn't run ("Package") or causes JVM exception (Extract)
B002 = 10:40PM (2/27/19) - opening image running to exception at 2nd selection (OpenImage.java)
B003 = 8:45PM (3/7/2019) - Camera disoriented when size at variable mode
B004 = N/A - Wrong Image Rescaling to Panel
==========
Glitches:

Official Project
G001 = 11:37PM (2/15/19) - GUI Components(buttons and output box) not displaying at initial runtime; hovering over the buttons fixes it temporarily; (FIXED)
==========
Fixes:
Fix for error in Huiying S. (2013)
#1 - highgui not supported in java = use "org.opencv.videoio.VideoCapture" instead of "import org.opencv.highgui.Highgui"
#2 - swap red and blue channels = Imgproc.cvtColor(mat,mat,Imgproc.COLOR_RGB2BGR); //to be placed at the getSpace() of Mat2Image

Official Project
11:37PM (2/15/19) - Viewport code redo; extends JFrame
6:59PM (2/16/19) - Viewport Jbuttons and Jtextarea repaint added to remove (G001)
10:04PM (2/18/19) - Fix found by pasting the dll file of opencv directly to the System32 on Windows, SetupDLL() disabled
10:49PM (2/27/19) - Fixed by adding a reset to the directory of files (B002)
*8:50PM (3/7/19) - Temporary fix at B003, no dynamic camera access, limited to 640x480 pixels (TF.java, Camera.java, Mat2Image.java)
3/17/19 - Image rescaling done, though not perfect,
==========
Changelog:
6:59PM (2/16/19) - Added popups on button interactions (Viewport.java)
10:04PM (2/18/19) - SetupDLL() disabled via commenting on connectCam() (Viewport.java)
7:08PM (2/21/19) - Added OpenImage class for file and io functions (OpenImage.java)
10:43AM (2/23/19) - Open Camera Button (cam_capt) functionality corrected; switches immediately to cam mode at click
8:34PM (2/25/19) - Added class for AI (Tensor.java)
8:34PM (2/25/19) - Added library diagnostics at terminal (Baybayin.java)
8:40PM (2/25/19) - Thread.sleep time changed from 100 to 80, whilst retaining GUI integrity
8:47PM (2/27/19) - Improved GUI response
10:26PM (2/27/19) - Added input of image (JPG)
10:52PM (2/27/19) - Enhanced image file operation workflow; added support for PNG.
~7-8PM (3/3/19) - Changed Thread.sleep time to 50
12:06AM (3/3/19) - Added comments on functions, added screen size based window sizing
8:51PM (3/7/19) - Revised image (Camera and Image) rendering workflow to passthrough TF for bound box rendering and application
9:14PM (3/7/19) - Tested boundbox rendering on Camera
9:21PM (3/7/19) - Tested boundbox rendering on Image
11:11AM (3/9/19) - Added Draw function on TF.java, where it draws boxes and string from a set of parameters; drawBox function removed from Viewport.java
11:56AM (3/9/19) - Added warnings when camera is not connected
5:45PM (3/9/19) - Code diet & added comments to most of the classes.
3/16/19-3/17/19 - Added kalas-puffar's code in Brain.java for reading class files; added co-execute code on py package (Out.java); added image in replacement for camera feed in camera mode, fixed scaling issue, corrected AreaAppend function and execution (Viewport.java)
4:28PM (3/18/19) - Reduced memory usage at start-up (~660MB -> ~330MB) and reduced RAM usage at runtime after detection through image file
8:25PM (3/18/19) - Improved execution timing of images, revision of Baybayin (C:\Baybayin) structure, fonts at array for selection in relation to screen resolution, location directories added as string on References.java, redirected java.library.path to C:\Baybayin\bin
9:10PM (3/18/19) - Added citations to code
5:12PM (3/27/19) - Deprecated Image function
==========
Notes:
1. OpenCV DLL architecture x86
2. TF DLL version GPU
3. Must have msvcp140.dll to run TF
==========

References (Code):
Huiying S. (2013). Retrieved from http://computervisionandjava.blogspot.com/2013/10/java-opencv-webcam.html
Solving Common Painting Problems. (n.d.). Retrieved from
https://docs.oracle.com/javase/tutorial/uiswing/painting/problems.html
Nuwan, D. (2016). Retrieved from http://answers.opencv.org/question/69226/error-when-lunching-the-runnable-jar-file/
How to Use FileChooser. (n.d.). Retrieved from
https://docs.oracle.com/javase/tutorial/uiswing/components/filechooser.html
Alexander, A. (2016). Retrieved from
https://alvinalexander.com/blog/post/java/java-faq-how-determine-java-version-application
kalaspuffar. (2018). Retrieved from https://github.com/kalaspuffar/rcnn-model-test
Display Image OpenCV Java. (n.d.). Retrieved from
https://riptutorial.com/opencv/example/25715/display-image-opencv-java
OpenCV - Reading Images. (n.d.). Retrieved from https://www.tutorialspoint.com/opencv/opencv_reading_images.htm
OpenCV - GUI. (n.d.). Retrieved from https://www.tutorialspoint.com/opencv/opencv_gui.htm
Alexander, A. (n.d.). Retrieved from
https://alvinalexander.com/blog/post/jfc-swing/how-determine-get-screen-size-java-swing-app
Persson, D. (2018). Object detection Part 6 - Inference in java [Tensorflow]. Retrieved from
https://www.youtube.com/watch?v=K0ewhYgKRpo&t=664s
Drawing Functions. (n.d.). Retrieved from
https://docs.opencv.org/3.0-beta/modules/imgproc/doc/drawing_functions.html
OpenCV - Adding Text. (n.d.). Retrieved from https://www.tutorialspoint.com/opencv/opencv_adding_text.htm
Draw rectangle on image. (n.d.). Retrieved from https://riptutorial.com/opencv/example/21372/draw-rectangle-on-image
Class Imgproc. (n.d.). Retrieved from https://docs.opencv.org/3.4/javadoc/org/opencv/imgproc/Imgproc.html
Nanjappa, A. (2013). Resize or rescale image in OpenCV. Retrieved from
https://gist.github.com/ashwin/565a6f8f09ce69367aab
Persson, D. (2018). RunInference.java. Retrieved from
https://github.com/kalaspuffar/rcnn-model-test/blob/master/src/main/java/org/ea/waldo/RunInference.java
How to Use File Choosers. (n.d.). Retrieved from
https://docs.oracle.com/javase/tutorial/uiswing/components/filechooser.html
FileChooserDemo.java. (n.d.) Retrieved from
https://docs.oracle.com/javase/tutorial/displayCode.html?code=https://docs.oracle.com/javase/tutorial/uiswing/examples/components/FileChooserDemoProject/src/components/FileChooserDemo.java
Create a BufferedImage from file and make it TYPE_INT_ARGB. (2012). Retrieved from
https://stackoverflow.com/questions/10391778/create-a-bufferedimage-from-file-and-make-it-type-int-argb
How to invoke a class in another package? [duplicate]. (2017). Retrieved from
https://stackoverflow.com/questions/40885553/how-to-invoke-a-class-in-another-package
Kim, Y. M. (2018). Retrieved from https://www.mkyong.com/java/java-processbuilder-examples/
Chaudhary, M. (2014). Java program to kill a runnning windows process. Retrieved from
https://gist.github.com/madan712/7246224
Gandhi, S. (2014). How to Convert Mat to BufferedImage & Vice Versa. Retrieved from
https://www.codeproject.com/Tips/752511/How-to-Convert-Mat-to-BufferedImage-Vice-Versa
Run batch file from Java code. (2014). Retrieved from
https://stackoverflow.com/questions/19103570/run-batch-file-from-java-code/19103784
Closing video window using "X" button in OpenCV, Python. Retrieved from
https://stackoverflow.com/questions/13307606/closing-video-window-using-close-x-button-in-opencv-python

References (Miscellaneous):
Installing OpenCV for Java. (n.d.). Retrieved from
https://opencv-java-tutorials.readthedocs.io/en/latest/01-installing-opencv-for-java.html
Install TensorFlow for Java. (n.d.). Retrieved from https://www.tensorflow.org/install/lang_java
Papp, D. (2018). https://stackoverflow.com/questions/48998772/setting-up-tensorflow-in-eclipse
Importing and Exporting Java Projects as JAR Files in Eclipse. (n.d.). Retrieved from
https://www.albany.edu/faculty/jmower/geog/gog692/ImportExportJARFiles.htm
Maxim. (2017). Retrieved from
https://stackoverflow.com/questions/47068709/your-cpu-supports-instructions-that-this-tensorflow-binary-was-not-compiled-to-u

References (Training):
Patil, R. (2018). How to use TensorFlow Object Detection API On Windows. Retrieved from
https://medium.com/@rohitrpatil/how-to-use-tensorflow-object-detection-api-on-windows-102ec8097699
protocolbuffers. Protocol Buffers v3.7.0. Retrieved from https://github.com/protocolbuffers/protobuf/releases/tag/v3.7.0
Juras, Evan. (2018). Retrieved from
https://github.com/EdjeElectronics/TensorFlow-Object-Detection-API-Tutorial-Train-Multiple-Objects-Windows-10
Tensorboard not connecting to port 6060 with Chrome. (2019). Retrieved from
https://github.com/EdjeElectronics/TensorFlow-Object-Detection-API-Tutorial-Train-Multiple-Objects-Windows-10/issues/219

Link to README.md is broken in NativeLibrary.java.(2017). Retrieved from https://github.com/tensorflow/tensorflow/issues/11015
TensorFlow cannot find the native library. (2017). Retrieved from https://stackoverflow.com/questions/43830660/tensorflow-cannot-find-the-native-library/47622203#47622203
Exception in thread “main” java.lang.UnsatisfiedLinkError: Cannot find TensorFlow native library for OS: linux, architecture: x86_64. (2018). Retrieved from https://stackoverflow.com/questions/46303270/exception-in-thread-main-java-lang-unsatisfiedlinkerror-cannot-find-tensorflo
Portrait Image Issues being treated as Landscape ones. (2017). https://github.com/tzutalin/labelImg/issues/159
JPEG Autorotate. (n.d.). Retrieved from https://savolai.net/software/JPEG-EXIF_autorotate
Installation (n.d.). Retrieved from https://github.com/tensorflow/models/blob/master/research/object_detection/g3doc/installation.md
Labinski, M. (2018). Installing TensorFlow Object Detection API on Windows 10. Retrieved from https://medium.com/@marklabinski/installing-tensorflow-object-detection-api-on-windows-10-7a4eb83e1e7b
Galarnyk, M. (2016). Install Python on Windows (Anaconda). Retrieved from https://medium.com/@GalarnykMichael/install-python-on-windows-anaconda-c63c7c3d1444
Tran, D. (2017). How to train your own Object Detector with TensorFlow’s Object Detector API. Retrieved from https://towardsdatascience.com/how-to-train-your-own-object-detector-with-tensorflows-object-detector-api-bec72ecfe1d9
error when generate tfrecord. (2018). Retrieved from https://github.com/datitran/raccoon_dataset/issues/41
Tran, D. (2018). Retrieved from https://github.com/datitran/raccoon_dataset
How to remove spaces from file names (in bulk). (2013). Retrieved from https://stackoverflow.com/questions/11270453/how-to-remove-spaces-from-file-names-in-bulk
No module named 'object_detection' (2017). Retrieved from https://github.com/tensorflow/models/issues/2031
Unable to generate tf records. (2017). Retrieved from https://github.com/datitran/raccoon_dataset/issues/14
ElifTech. (2018). Object Detecion with Tensorflow. Retrieved from https://www.slideshare.net/ElifTech/object-detection-with-tensorflow
Lazorenko, A. (2017). TensorFlow performanc test: CPU VS GPU. Retrieved from https://medium.com/@andriylazorenko/tensorflow-performance-test-cpu-vs-gpu-79fcd39170c
Majek, K. (2018). 10 simple steps to Tensorflow Object Detection API. Retrieved from https://medium.com/@karol_majek/10-simple-steps-to-tensorflow-object-detection-api-aa2e9b96dc94
Lake, B. (2019). Retrieved from https://github.com/brendenlake/omniglot
Lake, B. M., Salakhutdinov, R., & Tenenbaum, J. B. (2015). Human-level concept learning through probabilistic program induction. Science, 350(6266), 1332-1338.
Lake, B. M., Salakhutdinov, R., & Tenenbaum, J. B. (2019). The Omniglot Challenge: A 3-Year Progress Report. arXiv preprint arXiv:1902.03477.
I am having trouble learning Tensorflow. (2018). Retrieved from https://www.reddit.com/r/tensorflow/comments/97fll2/i_am_having_trouble_learning_tensorflow/

APPENDIX I: ADDITIONAL TECHNICAL INFORMATION AND ISSUES FOUND DURING DEVELOPMENT, TRAINING AND TESTING

A. **Why CUDA Toolkit 9.0?**
   a. **Key Solution:** Use CUDA Toolkit 9.0
   b. **Problem/Error:** Not working at CUDA Toolkit 10.0
   c. **Error occurrence:** Issue occurred at Appendix F
   d. **Reason:** "**TensorFlow officially supports CUDA 9.0.** However it is **compatible with CUDA 10.0 but not supported currently**. For using TF with cuda 10, you have to build it from sources yourself. You can also take a look at installations done by another users to make it work." -Github user: ymodak
   e. **Source:** https://github.com/tensorflow/tensorflow/issues/22872 by ymodak

B. **Why numpy 1.16.2?**
   a. **Error:** ModuleNotFoundError: No module named 'numpy.core._multiarray_umath'
   b. **Error occurrence:** Issue occurred at Appendix F
   c. **Key Solution:** Replace numpy with specific version (numpy 1.16.2)
   d. **Notes:** numpy v1.15 and above must work
   e. **Source:** https://github.com/alpacahq/pylivetrader/issues/73 by wishvivek

C. **System must have msvcp140.dll at runtime**
   a. **Notes:** Required by TensorFlow library at runtime
   b. **Locate:** "C:/Windows/System32/mvscp140.dll" if it exists, if not then refer to Key Solution
   c. **Key Solution:** Download and Install https://www.microsoft.com/en-us/download/details.aspx?id=48145
   d. **Source:** https://www.tensorflow.org/install/lang_java#run

D. **CPU with AVX (Advanced Vector Extensions) instruction set possibly required to run**
   a. **Error:** Exception in thread "main" java.lang.UnsatisfiedLinkError: Cannot find TensorFlow native library for OS: windows, architecture: x86_64.
   b. **Error occurrence:** At runtime
   c. **Key Solution:** Limit Application Compatibility to CPUs with AVX Instruction Set
   d. **Notes:** The test was done on systems of five different CPUs.
   e. **Testing:** The test conducted was based from an installation guide of TensorFlow for Java

	CPU 1	**CPU 2**	**CPU 3**	**CPU 4**	**CPU 5**
Brand	Intel	Intel	AMD	Intel	Intel
Model	Core 2 Duo	i3	Ryzen 3	Celeron	i7
Variant	E7300	2100	1200	3000 series (specific	6700HQ

				variant is unknown)	
Year of Production	2008	2011	2017	2016	2015
Cores	2	2	4	N/A	4
Threads	2	4	4	N/A	8
Frequency	2.66Ghz (OC to 3.2Ghz)	3.1Ghz	3.1Ghz	N/A	2.6Ghz
Support for AVX	No	Yes	Yes	No	Yes
Support for AVX 2	No	Yes	Yes	No	Yes

However a possible isolated case occurred on the system with CPU 5 where the issue still occurred, but it was determined that msvcp140.dll may have been non-existent in Windows which is required by TensorFlow to run.

E. Source:

a. https://github.com/tensorflow/tensorflow/issues/20563
b. https://github.com/tensorflow/tensorflow/issues/19584
c. https://github.com/tensorflow/tensorflow/issues/18530
d. https://www.tensorflow.org/install/lang_java

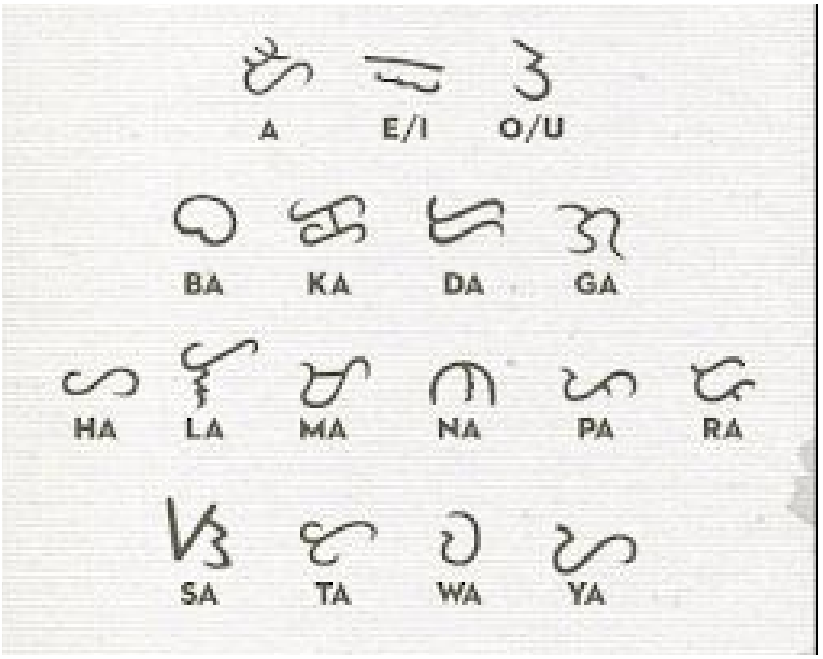
F. Fix for possible error in generate_tfrecord.py

a. Error occurrence: Whenever launched
b. Fix: In the end of the if-else statements at the else portion, change “return None” to “return 0” as the value requests for an integer instead of a statement.

G. object_detection_webcam.py returns N/A as interpretation:

a. **Fix:** Update number of classes in relation to the number of classes
b. **Source:** https://github.com/EdjeElectronics/TensorFlow-Object-Detection-API-Tutorial-Train-Multiple-Objects-Windows-10/issues/46

APPENDIX J: BAYBAYIN REFERENCE



Source:

Lahi.PH. (2018). Mahirap ba matuto ng Baybayin? [Facebook Video Post]. Retrieved from https://www.facebook.com/Lahi.PH/videos/417297688813792/?v=417297688813792

APPENDIX K: BAYBAYIN AI - DEVELOPMENT REPORT

- **Current Version:** 0.11.10
- **Date Created:** March 20, 2019
- **Features:**
    - Camera Integration via Python Co-execution
    - TensorFlow in Python and Java
- **Installation Requirements:**
    - Executable Jar File
    - Python (pip installed modules)
        - tensorflow-gpu (v1.12.0)
        - pandas
        - pillow
        - lxml
        - cython
        - jupyter
        - matplotlib
        - pandas
        - Opencv-python
    - JDK or JRE 1.8
    - CuDNN
    - CUDA Development Kit
    - **Runtime Files:**
        - **Notes:**
            - To be located at C:\
            - Strictly follow this since files are needed to run
        - Some files are from the training folder (D:\tensorflow1\models\research\object_detection\inference_graph)
        - **File Structure:**
            - C:
                - Baybayin
                    - __pycache__
                    - assets
                        - cameramode_load.jpg
                        - cameramode.jpg
                    - export
                        - saved_model
                        - checkpoint
                    - training
                    - utils
                    - camera
                    - frozen_inference_graph.pb
                    - label_map_util.py
                    - labelmap.pbtxt
                    - run.bat
                    - saved_model.pb
                    - visualization_utils.py
- **Software & Hardware Limitations/Specifications:**
    - Requires a Nvidia GPU to run efficiently
    - Requires 4GB RAM
    - It can run on CPU but not on Camera Mode
    - JAR file size: 2.06MB

    - Runtime files (Baybayin folder): 100MB (as of 3-17-19)
- **AI Limitations:**
    - (as of 3/17/19)
        - Inaccurate detections due to the following:
        - Lighting
        - Insufficient training data
        - Background
        - Distance
    - Camera must be positioned at least 1 foot from the sample
    - Image must be captured at least 1 foot from the sample

From developer's point of view

System Related Recommendations:

1. Improve program efficiency
2. Improve GUI design and responsiveness to be more user friendly
3. Embed assets and other related files within the app or a predetermined directory upon applications installation thru a proper installer
4. Implement this in a mobile version, thus a use of a server is also suggested to conduct inferencing which is similar to the way Google Translate is implemented aside from local inferencing to improve performance, efficiency and accuracy.
5. Use other programming languages apart from Java for increased API support:
    a. Languages currently supported by TensorFlow:
        i. Android Java (most recommended for mobile due to extensive API support)
        ii. Java
        iii. Swift
        iv. Go
        v. Haskell
        vi. Javascript
        vii. C++
        viii. Python (most recommended due to extensive API support)
    b. Source: https://www.tensorflow.org/api_docs
6. AI Related Recommendations:
    a. Train on a system with higher specifications
        i. Possible solutions:
            1. Use of a server with more CPU cores, more RAM and/or better GPU
            2. Use of a TPU (Tensor Processing Unit) as a training accelerator
            3. Use of a cloud service for training for increased performance and reliability
            4. Reason: Due to RAM (CPU) and VRAM (GPU) usage during training resulting to slow training or eventual crashing
    b. Use more datasets with the following parameters
        i. Different lighting
        ii. Diverse fonts/handwritings
        iii. Note: Characters must be strictly upright since it is a reading character
7. Baybayin Recommendations:
    a. Ensure that the data used was academically accepted set of Baybayin Symbols
    b. Include more Baybayin Symbols (Modern Baybayin)

APPENDIX L: PROTOBUF (PROTOC) USAGE

Protobuf (protoc) download URL:
https://github.com/protocolbuffers/protobuf/releases
Choose:
protoc-x.x.x-win64.zip or protoc-x.x.x-win32.zip
in this case - 3.7.0-win64
Unzip and place on:
C:\Program Files for x64 or C:\Program Files (x86) for (x86)
When doing 2.f from:
https://github.com/EdjeElectronics/TensorFlow-Object-Detection-API-Tutorial-Train-Multiple-Objects-Windows-10
Add:
"C:\Program Files\protoc-3.7.0-win64\bin\protoc.exe" first and remove protoc before the proceeding commands
via cmd:
"C:\Program Files\protoc-3.7.0-win64\bin\protoc.exe" --python_out=.
.\object_detection\protos\anchor_generator.proto
.\object_detection\protos\argmax_matcher.proto
.\object_detection\protos\bipartite_matcher.proto
.\object_detection\protos\box_coder.proto .\object_detection\protos\box_predictor.proto
.\object_detection\protos\eval.proto .\object_detection\protos\faster_rcnn.proto
.\object_detection\protos\faster_rcnn_box_coder.proto
.\object_detection\protos\grid_anchor_generator.proto
.\object_detection\protos\hyperparams.proto
.\object_detection\protos\image_resizer.proto
.\object_detection\protos\input_reader.proto .\object_detection\protos\losses.proto
.\object_detection\protos\matcher.proto
.\object_detection\protos\mean_stddev_box_coder.proto
.\object_detection\protos\model.proto .\object_detection\protos\optimizer.proto
.\object_detection\protos\pipeline.proto .\object_detection\protos\post_processing.proto
.\object_detection\protos\preprocessor.proto
.\object_detection\protos\region_similarity_calculator.proto
.\object_detection\protos\square_box_coder.proto .\object_detection\protos\ssd.proto
.\object_detection\protos\ssd_anchor_generator.proto
.\object_detection\protos\string_int_label_map.proto
.\object_detection\protos\train.proto .\object_detection\protos\keypoint_box_coder.proto
.\object_detection\protos\multiscale_anchor_generator.proto
.\object_detection\protos\graph_rewriter.proto
Source:

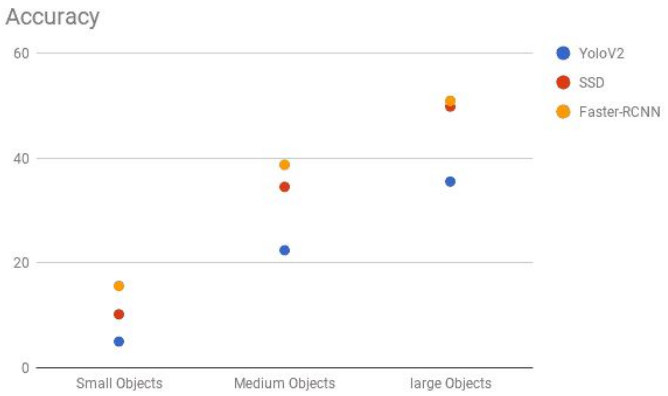
- https://stackoverflow.com/questions/52929161/cannot-find-protoc-command?noredirect=1&lq=1
- https://github.com/EdjeElectronics/TensorFlow-Object-Detection-API-Tutorial-Train-Multiple-Objects-Windows-10

Note:
Or you can set as path: protoc to the bin folder of protoc to make it universal

APPENDIX M: BEST TRAINING RESULT

Based from the "Zero to Hero: Guide to Object Detection using Deep Learning: Faster R-CNN,YOLO,SSD" (n.d.), the best possible method of object detection model or method is Faster R-CNN which based from different tests showed that it was the fastest and the most accurate.

Accuracy and speed comparison of different Object Detection Models/Methods



Accuracy of different Object Detection Models/Methods on objects of different size

The data shown below are the configuration and result data of the training with the best possible accuracy. The training configuration are as follows:

Component	**Value**
Machine Learning Model	Faster R-CNN
Feature Extractor	Inception v2
Pre-trained dataset	Common Objects in Context (COCO)
Number of classes	17
Number of Images in Training Folder	355
Number of Images in Testing Folder	59

Training Configuration

Loss, is simply a number that indicates to how bad is the model's prediction on a given example ("Descending into ML: Training and Loss", n.d.). The closer the value is to zero, the better is the model's prediction. The training's target is to achieve a low loss value to ensure good predictions and the following table shows that the researchers have achieved a relative good loss value given the training steps and training data utilized, however this is still must be double checked for reasons that loss values can be at times

Component	**Value**
Total Training Steps	67,716
Classification Loss (BoxClassifierLoss)	0.1640
Localization Loss (BoxClassifierLoss)	0.1072
Total Loss	0.3128

Training Results

Source:

Descending into ML: Training and Loss. (n.d.). Retrieved from https://developers.google.com/machine-learning/crash-course/descending-into-ml/training-and-loss

Zero to Hero: Guide to Object Detection using Deep Learning: Faster R-CNN,YOLO,SSD. (n.d.). Retrieved from https://cv-tricks.com/object-detection/faster-r-cnn-yolo-ssd/

APPENDIX N - SURVEY TALLY AND COMPUTATION

Respondent	Percent	Q1	Q2	Q3	Q4	Q5	Q6	Q7
1	99	3	3	2	3	3	4	5
2	99	4	4	3	4	4	4	3
3	99	5	5	5	5	5	5	5
4	0	2	4	4	3	5	4	4
5	98	4	4	5	3	5	5	5
6	90	5	5	5	5	5	5	5
7	95	5	5	5	5	5	5	5
8	80	4	5	5	5	5	5	5
9	95	5	4	5	4	5	4	5
10	98	5	5	5	5	5	5	5
11	99	4	5	4	5	5	5	5
12	99	5	5	4	4	5	5	4
13	99	5	5	5	5	5	5	5
14	80	4	4	5	3	4	4	4
15	98	4	5	3	4	3	3	3
16	80	4	4	4	4	4	3	5
17	90	5	5	5	5	5	5	5
18	99	5	5	5	5	5	5	5
19	99	5	5	5	5	5	5	5
20	80	4	4	3	4	4	5	5
21	90	5	5	4	4	4	4	5
22	0	4	4	3	4	3	4	4
23	0	4	4	4	4	4	4	4
24	90	5	5	5	5	5	5	5
25	78	4	4	4	3	3	4	4
26	99	5	5	4	5	4	5	5
27	97	5	5	5	4	5	5	5
28	98	5	5	4	4	5	5	5
29	97	5	5	5	4	5	5	5
30	0	3	3	4	4	3	3	4
31	91	5	5	5	5	5	5	5
32	0	2	2	2	2	2	2	2
33	96	5	5	5	5	5	5	5
34	98	5	5	5	5	5	5	5
35	98	4	5	4	5	3	4	4
36	89	5	4	5	5	5	5	4
37	79	3	4	3	3	4	3	4
38	90	5	5	5	5	5	5	5
39	70	5	5	5	5	5	5	5
40	97	5	5	5	5	5	5	5
41	0	5	5	5	5	5	5	5

42	85	5	5	5	5	5	5	5
43	0	5	5	5	5	5	5	5
44	0	5	5	5	5	5	4	5
45	99	5	4	5	5	5	4	4
46	88	5	5	5	5	5	5	5
47	99	5	5	4	5	4	4	5
48	99	5	5	5	5	5	5	5
49	90	5	5	5	5	5	5	5
50	95	5	5	5	5	5	5	5
MEAN		4.52	4.6	4.44	4.44	4.52	4.52	4.64

APPENDIX O - ACCURACY TEST

Letter	Attempts	1	2	3	4	5	Success Quantity	Rate	Accuracy
A	5	1	1	1	1	1	5	100.00%	Accurate
E/I	5	0	1	1	1	1	4	80.00%	Accurate
O/U	5	0	0	0	0	0	0	0.00%	Not Accurate
Ba	5	1	1	1	1	1	5	100.00%	Accurate
Da	5	1	1	1	0	1	4	80.00%	Accurate
Ka	5	1	1	1	1	1	5	100.00%	Accurate
La	5	0	1	1	0	1	3	60.00%	Not Accurate
Ma	5	1	1	1	1	1	5	100.00%	Accurate
Na	5	1	1	1	1	1	5	100.00%	Accurate
Pa	5	0	0	0	0	0	0	0.00%	Not Accurate
Sa	5	1	1	1	1	1	5	100.00%	Accurate
Ta	5	1	1	1	1	1	5	100.00%	Accurate
Wa	5	1	1	1	1	0	4	80.00%	Accurate
Ya	5	1	0	0	1	1	3	60.00%	Not Accurate
Ga	5	1	1	1	0	1	4	80.00%	Accurate
Ha	5	1	1	1	1	1	5	100.00%	Accurate
Ra	5	0	0	0	0	0	0	0.00%	Not Accurate

APPENDIX P - REQUEST LETTER

ELIZABETH SETON SCHOOL – SOUTH
Anabu II-D, Imus City, Cavite 4103

Angelo P. Profeta
Owner

We the researchers of the Capstone Project: TRANSLATION OF BAYBAYIN CHARACTERS THROUGH ARTIFICIAL INTELLIGENCE would like to ask permission to borrow your laptop on the April 1st and 2nd of 2019 for last minute testing and demonstration during the Capstone Defense.

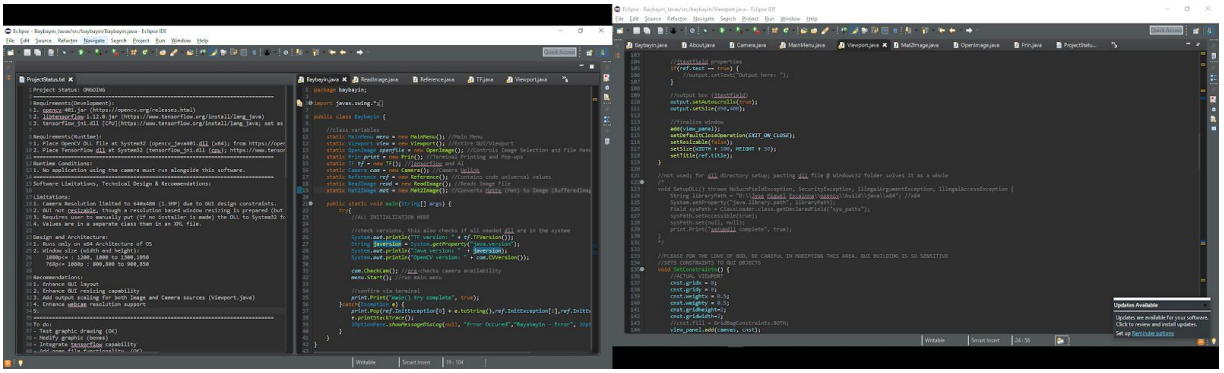
Sign when Permission Granted: _____

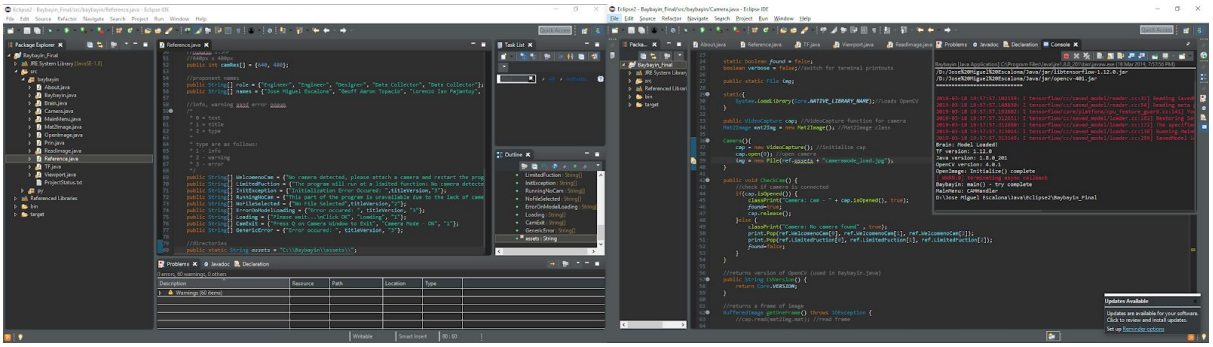
Sincerely,

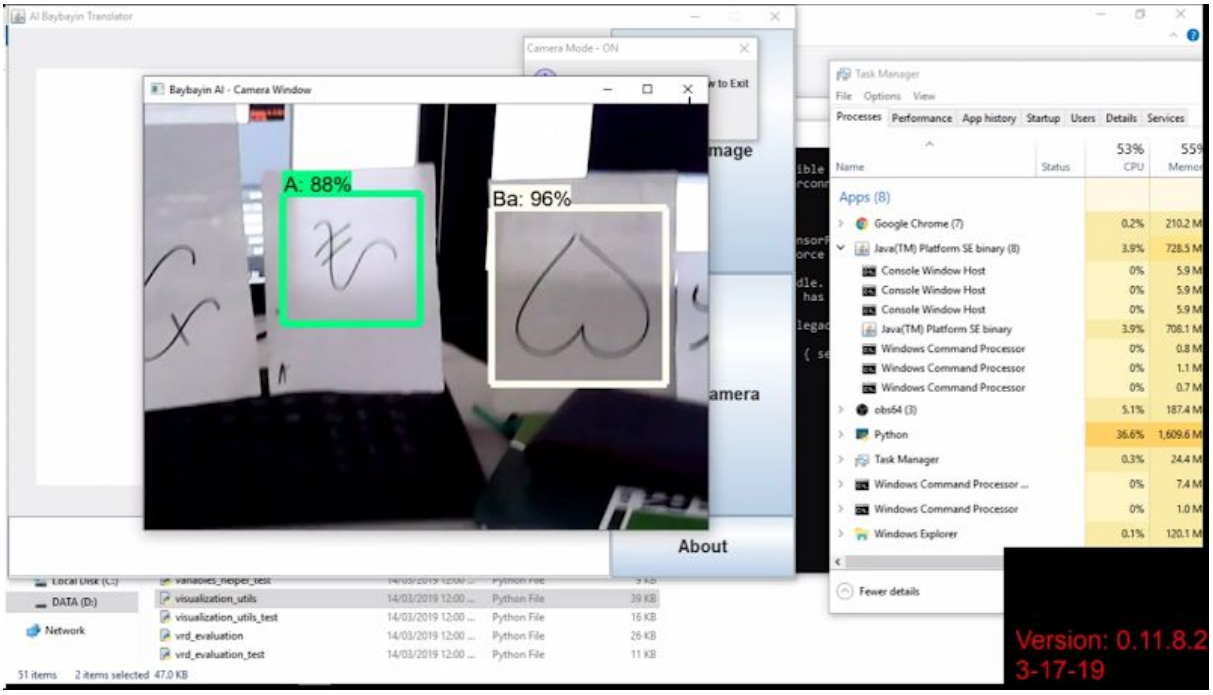
Francis Bert L. Dolot Jose Miguel A. Escalona Lorenzo Ian A. Pajantoy

Dean M. Reyes Geoff Aaron C. Topacio

APPENDIX R - DEVELOPMENT AND RUNTIME SCREENSHOTS







CURRICULUM VITAE

Francis Bert L. Dolot

Blk 9 Lot 14 Aston Martin Street,

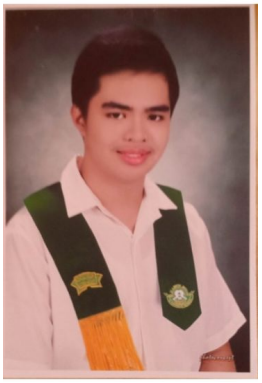
Citihomes Grand Plaza,

Anabu II-A,

Imus City, Cavite

09054576955

dolot.francis@ess.edu.ph



Personal Information

Age: 17

Civil Status: Single

Nationality: Filipino

Religion: Catholic

Language: Filipino & English

Gender: Male

Positions Held

N/A

Education Attainment

Elizabeth Seton School-South – Grade 11 Senior High School(2017 - Present)

Elizabeth Seton School-South – Grade 7-10 Junior High School(2013 -2017)

Elizabeth Seton School-South – Grade 4-6 Elementary(2010 - 2013)

Casa De San Miguel Montessori – Grade 3 Elementary(2009 - 2010)

Seiko Grandioso Montessori – Grade 2 Elementary(2008 - 2009)

Asiana Italia Montessori – Grade 1 Elementary(2007- 2008)

Mabolo Elementary School – Preschool(2004 - 2007)

Award Received

3rd Philippine Robothon: Expert Category – participant(2014)

Professional Organization

House of St. John XXIII – member

The Climax – member

Seminars Attended

Elizabeth Seton School - South: Career Week(2017)

Hour of Code(2016)

Seminars Conducted

Hour of Code(2016)

Action Research Conducted

Evaluation in the knowledge of Microsoft Powerpoint of the Senior High School Students of Elizabeth Seton School - South(2018)

The Impact of Social Media on the Political Stances of JHS Students from Grades IX-X (2017)

Jose Miguel A. Escalona

Blk 8 Lt 12 Campanilla St.

Area-D Parkplace Village,

Anabu I-D,

Imus City, Cavite

09275865581

escalonaschool@gmail.com

Personal Information

Age: 18 years old

Civil Status: Single

Nationality: Filipino

Religion: Roman Catholic

Language: English & Filipino

Gender: Male

Positions Held

Film Editor – Philosophical Film Festival Entry: Excess (2019)

Film Editor – Only I (2018)

Auditorium Technical Director – 10-Noveleta: Les Miserable Play (2017)

Finance Manager – 10-Noveleta: Business Simulation (3rd Term, 2017)

Finance Manager – 10-Noveleta: Business Simulation (2nd Term, 2016)

Finance Manager – 10-Noveleta: Business Simulation (1st Term, 2016)

Educational Attainment

Elizabeth Seton School-South – Grade 12: Senior High School (2018-2019)

Elizabeth Seton School-South – Grade 11: Senior High School (2017-2018)

Elizabeth Seton School-South – Grade 7-10: Junior High School (2013-2017)

Elizabeth Seton School-South – Grade 1-6: Elementary (2007-2013)

Casa Real Montessori – Preparatory (2006-2007)

Statefields School – Nursery-Kinder (2004-2006)

Awards received

Participation in MCL Cup – Photoshop Competition (2017)

Participation in DISCS Programming Competition – Ateneo University Diliman (2017)

Professional Organization

House of St. Paul - Member

SiningGang Club - Member

Seminars Attended

ESS-South: Career Week (2017)

Hour of Code (2016)

Seminars Conducted

Hour of Code (2016)

Action Research Conducted

Evaluation in the knowledge of Microsoft PowerPoint of the Senior High School Students of Elizabeth Seton School – South (2018)

Allowance received by grade 10 students and their ability to save (2017)

Lorenzo Ian A. Pajantoy

B8 L24, Dao Street,

Villa Luisa Homes,

Dasmarinas Cavite

0927-501-9688

lorenzopajantoy@gmail.com

Personal Information

Age: 18 years old

Civil Status: Single

Nationality: Filipino

Religion: Roman Catholic

Language: English & Filipino

Gender: Male

Positions Held

● House of St. Peter	Member
● Robothon Team	Builder
● Seton Notes	Editorial
● Sining Gang (Arts, Music, Design Club)	Editor

Educational Attainment

Senior High School (Grade 11 – 12): Elizabeth Seton School – South (S.Y. 2017 - 2019)

- Competed at the Malayan Digital Poster Making (Grade 11)

Junior High School (Grade 7 – 10): Elizabeth Seton School – South (S.Y. 2013 - 2017)

- 1st runner up Table Tennis Singles A (Grade 7)
- Competed on Robothon (Robotics Competition) at San Beda (Grade 8)
- Competed on Robothon (Robotics Competition) at La Salle Laguna (Grade 9)
- Champion at Robotics (Sumo Bot) Intramurals (Grade 10)
- Saint Peter Intramurals Team Captain (Grade 10)

Primary Education (Grade 1 – 6): Vel Maris School (S.Y. 2007 – 2009)/ Elizabeth Seton School – South (S.Y. 2009 – 2013)

- Attended the gathering of scouts held by Imus Pilot (Grade 4)
- Joined the CCP performance (Grade 4)
- Performed “Sabayang Pagbigkas” (Grade 6)

Preschool (Nursery – Preparatory): Sta. Belina Learning School / Vel Maris School (S.Y. 2004 - 2007)

Awards received

Participation in MCL Cup – Photoshop Competition (2017)

4th Philippine Robothon – Participation, October 24, 2015

5th Philippine Robothon – Participation, October 2015

Professional Organization

House of St. Peter - Member

SiningGang Club - Member

Seminars Attended

- Paths (July 2014) ESS- SOUTH Auditorium
- Anti- Bullying (2014) ESS- SOUTH Auditorium
- SHS Orientation (2015) ESS- SOUTH Auditorium
- Choices (Aug 2016) ESS- SOUTH Auditorium
- Alumni Symposium (July 2015) ESS- SOUTH Auditorium
- Team Building (Oct 10 2015) ESS- SOUTH Auditorium
- Career Week (Aug 2016) ESS- SOUTH Auditorium
- School Talk (September 6 2017)

Seminars Conducted

Hour of Code (2016)

Action Research Conducted

The Beneficial Effect of Utilizing Smartphones in the Classroom Among Grade 12 Students of Elizabeth Seton School Campuses (2018)

Dean Andre M. Reyes

Blk 8 Lot 12 Good Family Homes

Anabu 1-A

Imus City, Cavite

(0935)-228-3868

deanreyes30@gmail.com

Personal Information

Age: 18 years old — Date of Birth: Dec 30, 2000

Gender: Male — Religion: Roman Catholic

Language/s Spoken: English & Filipino — Nationality: Filipino

Civil Status: Single

Educational Background

Senior High School (2017 - present)

Elizabeth Seton School - South (Grade 11 – Grade 12)

Anabu II-D Imus City, Cavite

Junior High School (2013-2017)

Elizabeth Seton School - South (Grade 7 - Grade 10)

Anabu II-D Imus City, Cavite

Grade School (2005-2012)

· Jesus Good Shepherd School (Grade 1 – Grade 6)

Palico IV Imus City, Cavite

Pre-School (2004-2005)

· Jesus Good Shepherd School (Junior Kinder – Senior Kinder)

Palico IV Imus City, Cavite

Pre-School (2003)

Day Care Center, Good Family Homes Anabu 1-A

Professional Organization

- JGSS Volleyball Varsity Member (2012)
- Basketball Club Member (2010 - 2012)
- The Climax Club Member (2017 - 2018)
- House of St. Paul Member (2013 - 2019)

Seminars Attended

Paths (July 2014) - ESS- South Auditorium
Anti- Bullying (August 2014) - ESS- South Auditorium
SHS Orientation (June 2015) - ESS- South Auditorium
Team Building (October 2015) - ESS- South Auditorium
Career Week (August 2016) - ESS- South Auditorium
Hour of Code (December 2016) - ESS- South Auditorium

Awards Received and Achievements

Grade 11 (2016- 2017)

Champion Volleyball Boys Intrams

Grade 10 (2016- 2017)

Off –Campus

Basketball 3rd Place Sports League (Subdivision)

In – Campus

Champion Volleyball Boys Intrams

Grade 9 (2015 – 2016)

Champion Volleyball Boys Intrams

Grade 8 (2014 – 2015)

Champion Volleyball Boys Intrams

Character Reference

Vilma Reyes Gerard Godoy

Vince Aguiling Dante Vargas

Ms. Carmencita Victa

Geoff Aaron Topacio

Blk 6 Lt 4 Hausland Subdivision,

Anabu 1-B,

Imus City, Cavite

09239120893

topacio.geoff@ess.edu.ph

Personal Information

Age: 18 years old

Civil Status: Single

Nationality: Filipino

Religion: Roman Catholic

Language: English & Filipino

Gender: Male

Positions Held

Sining Gang - Member

House of St. John XXIII - Member

Educational Attainment

Elizabeth Seton School-South – Grade 12: Senior High School (2018-2019)

Elizabeth Seton School-South – Grade 11: Senior High School (2017-2018)

Elizabeth Seton School-South – Grade 7-10: Junior High School (2013-2017)

Elizabeth Seton School-South – Grade 2-6: Elementary (2008-2013)

Cavite School of St. Mark - Grade 1: Elementary (2007-2018)

Logic Gestalt Integrated Learning School – Kindergarten - Preparatory (2005-2007)

Joy Learning School – Nursery (2004-2005)

Awards received

Bronze Medalist (Grade 11, 2017 - 2018)

MCL Gastronomic Challenge (Participation) – September 13, 2017

Silver Medalist (Grade 10, 2016 - 2017)

Best in Computer (Grade 10, 2016 - 2017)

DISCS Programming Opening (Participation) – February 2017

DLSU Spark 2016 – 3rd Place, April 8, 2016

DISCS Programming Open (Participation) – February 2016

CAL 2015 Battle of ICT Wizards (Nationals) – Participation, February 12, 2016

CAL 2015 Battle of ICT Wizards (Regionals) – 2nd Place, January 3, 2016

4th Philippine Robothon – Participation, October 24, 2015

UP Circle of Entrepreneurs Young Entrepreneurs Convention: Enable – Participation, October 10, 2015

Professional Organization

House of St. John XXIII - Member

SiningGang Club - Member

Imus Youth - City IT Officer

Seminars Attended

ESS-South: Career Week (2017)

Hour of Code (2016)

IYLDP (2016)

SPARK (2016)

UP Circle of Entrepreneurs Young Entrepreneurs Convention: Enable – October 10, 2015

Seminars Conducted

Hour of Code (2016)

Action Research Conducted

The Usability of Different Smartphones According to its Users (2019)

Evaluation in the knowledge of Microsoft PowerPoint of the Senior High School Students of Elizabeth Seton School – South (2018)

Effectiveness of Different Teaching Styles in Teaching Mathematics 9 in Elizabeth Seton School - South S.Y. 2016-2017 (2017)