Faster Transcription Using a New Shorthand and Machine Learning

Ronit Avadhuta
Massachusetts Academy of Math and Science
STEM 1
Dr. Kevin Crowthers
2/5/2021

Contents

Abstract

Society demands efficiency yet typing on the standard keyboard can prove detrimental in fast-paced environments. This project aims to digitally transcribe information more efficiently, by interpreting and translating a new, custom, and efficient shorthand, a way of quickly writing English text using loops, curls, and squiggles, via Machine Learning image recognition software. Omer Alptekin, a former classmate, helped in the creation of this shorthand.

The first step in building the system is to transcribe common phonetic texts into the shorthand, take pictures, and annotate characters using the program LabelImg built specifically for this task. About 80% of these pictures and their xml files denoting character locations are the training set used to train the Tensorflow Recursive Neural Network. After training for 160,000 steps, the model is run with some code and a UI to add context to the words written. The total loss (inaccuracy of the Machine Learning model in identifying characters) was 0.11 to 0.12. Although the loss was not zero, the system provides a quick and easy way to communicate information through this model majority of the time. It provides a quantitative metric of the effectiveness of the trained model and a new medium for faster digital transcription for everyone in all environments. The digital shorthand system can help people save time writing on touchscreen devices such as iPads, write without the traditional keyboard (easier for the visually impaired), and standardize or facilitate shorthands in the medical industry.

Literature Review

Society demands efficiency. This is evident with several recent inventions, such as 5G and online medical treatment. However, while text messages can be sent in less than 5 seconds, society still types on the same QWERTY keyboard used in 1874. Society is not limited by the speed technology works but rather at the speed people type. Using specialized equipment such as a Steno, people can reach speeds of 360 words per minute (wpm) to finally meet normal speaking pace (MacMillan, n.d.), but not everyone has access to such specialized equipment. On computers they can only reach 52 wpm or 38 wpm on phones (Touchscreen Typing Speeds Close in on Keyboard Rates - BBC News, 2019). As part of the Information Age, it is time to look for a new, more accessible solution to data input.

Developing a New Shorthand

Shorthands are a quick way to denote textual information in a few short scribbles, an engineering marvel for information density. Although there are many shorthands, Gregg Shorthand is widely considered to be one of the world's best shorthands. However, it was created over a hundred years ago in a different era with different technology. It has also lost a great following with the advent of computers and other electronic devices (Rajasekaran, 2014). Thus, it would be fair to say Gregg Shorthand (the world's current leading option for the fastest writing convention) was not enough to satisfy the needs of today's world. The world needs a simpler, more efficient solution. Thus, it is hypothesized that it should be possible to create a new shorthand that utilizes less strokes and shorter strokes than Gregg Shorthand, the current leader, to minimize writing time. My work last year established that it was indeed possible to create a new shorthand utilizing less strokes and pixels: Experimental Group 3 or otherwise known as the Digital Shorthand Key. From here, it is possible to look towards digital storage of this information since a lower variety of characters requires less information to encode each character which leads to faster processing and input speeds. This appears to be a particularly lucrative endeavor since each character in the shorthand takes less time to write and there are fewer characters, making it more efficient and concise for existing widespread systems such as Unicode. By representing the same characters on paper with digital values, it may be possible to store data more efficiently thanks to the cutting the variety of characters almost in half, thus saving more space and processing time. Furthermore, with faster processing speeds, it could help with a large variety of tasks that include everything from self-driving cars to processing online purchases faster.

The Shorthand in Context

This year, the shorthand will be put to use when answering the following question: How can a faster method for data input be found that does not need specialized equipment, such as a steno, and is accessible to everyone? To better understand this year's project, some background knowledge on languages in general, then shorthands and machine learning would be beneficial along with a short description of my project last year and prior research.

Making Writing More Efficient

Foremost, it is important to note that written language in general has some areas for improvement. For example, one article focuses on drawing conclusions between 17 languages across 9 language families to demonstrate the relationships between speech, information density, number of syllables etc (Different Languages, Similar Encoding Efficiency: Comparable Information Rates across the Human Communicative Niche | Science Advances, n.d.). The project had seemingly taken into consideration several variables such as sex and language family that could have contributed to different rates in information and speech. After several mathematical tests and syntagmatic analysis, Figure 1 showed that languages varied greatly in speech while they were remarkably similar in Information rate. This therefore implies the notion that there are different efficiencies for different writing systems. This is important to note as it allows researchers to find similarities between languages of low speech rates and a high information density. This in turn helps researchers pinpoint which languages pack the most information into a singular phoneme and have similarities in their writing systems. The notion that there are similarities between the most efficient languages helps researchers create models for faster methods of communication. Information rate was defined in the experiment as the product of the speech rate and information density.

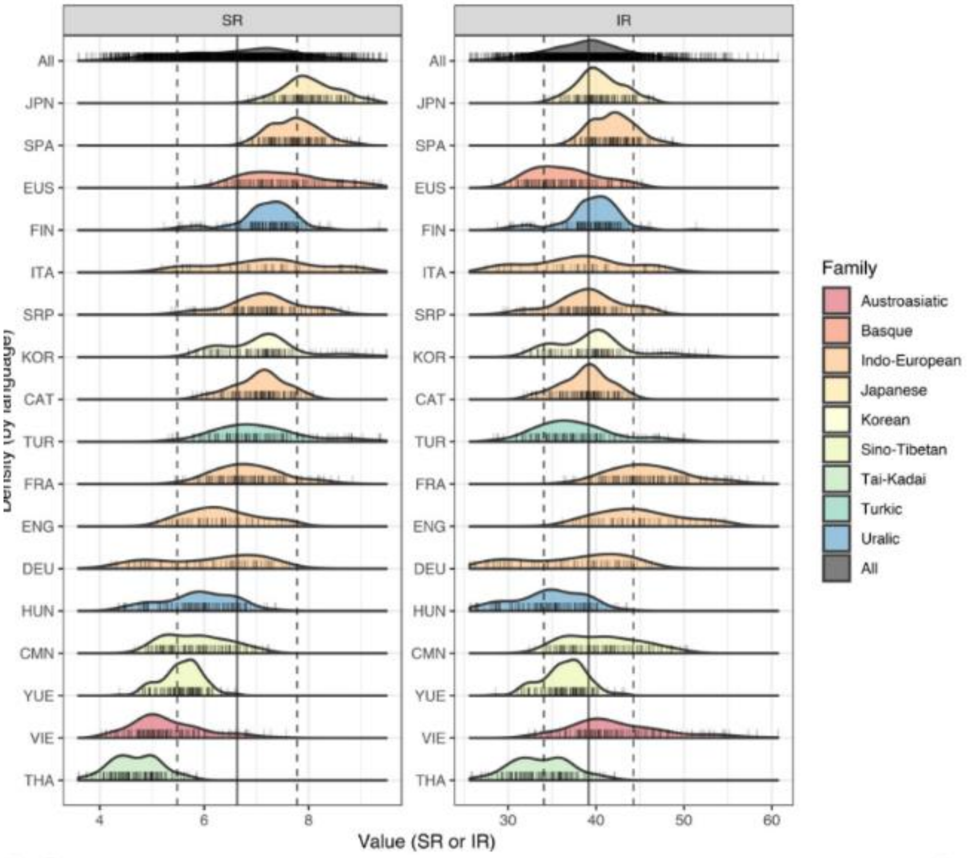
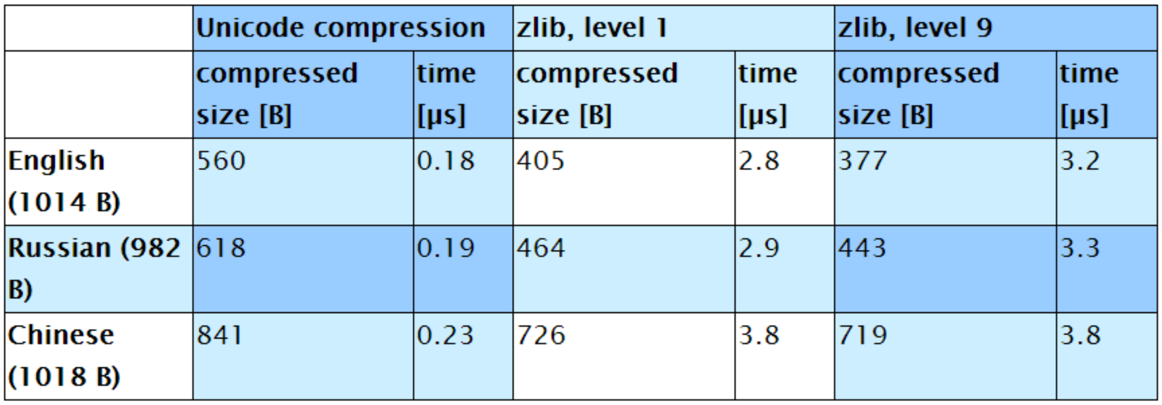


Figure 1. Speech Rate (SR) and Information Rate (IR) across Languages

The Digital World

The notion that there are characteristics of languages that make them more efficient than others can not only be seen in languages, but also be seen in a Unicode article on zip files (Fast Compression Algorithm for UNICODE Text, n.d.). Compression and decompression, such as is done with zip files, are nothing but a matter of assigning values to different segments of the text. One such method for this process is Lempel-Ziv compression that takes previous segments, which may be beneficial depending on repetitiveness of the sample. Then they also use a hash table which is a method almost identical to zip files. However, their article also goes into detail about the storage space of the characters. There are also some same algorithms and times (in microseconds) for how long this compression and the process should take. Looking at the following figure, a significant gap between languages such as from Chinese to English (22% decrease from 0.23 milliseconds to 0.18 milliseconds) illustrates the fact that repetition varies from language to language and highlights important languages and areas of interest. Furthermore, the characters or words highly repeated can be prioritized in order of frequency in order to take a highly effective approach at transcription.



	Unicode compression		zlib, level 1		zlib, level 9	
	compressed size [B]	time [µs]	compressed size [B]	time [µs]	compressed size [B]	time [µs]
English (1014 B)	560	0.18	405	2.8	377	3.2
Russian (982 B)	618	0.19	464	2.9	443	3.3
Chinese (1018 B)	841	0.23	726	3.8	719	3.8

Table 1. Unicode Built-In Compression Rate Across 3 Languages

But Natural Language Processing (NLP) itself is a bit different. It deals more with the syntactical and semantic sides of language and is thus a more complicated process that is case dependent and largely prone to error compared to Unicode compression. This field includes many types of models and techniques ranging from Hidden Markov Models to simple Backwards Propagation (Nadkarni et al., n.d.). NLP has come a long way and is even embodied in modern inventions such as IBM’s Watson. This leads to a discussion of its application in this year’s project.

Machine Learning for Text

The output of the first Machine Learning program will output information written in the DSK (Digital Shorthand Key), however this is not English but a representation of the 12 characters in the shorthand. To get it back into English, lexers and parsers will have to be used, or another Machine Learning algorithm that strictly deals with text to put the information into context (US7027974B1 - Ontology-Based Parser for Natural Language Processing - Google Patents, n.d.). Furthermore, although the information about English words in IPA is phonetic like the shorthand, the International Phonetic Alphabet will have to go through a dichotomy to resemble the DSK.

IPA (International Phonetic Alphabet) in conjunction with Unicode

There have been a number of groups pairing IPA with Unicode. The solution of one such group was a phonetic-based text input method for a patent that also focuses on the phonetic aspect of language to translate ideas between different mediums (US8200475B2 - Phonetic-Based Text Input Method - Google Patents, n.d.). The patent details a system where a program accurately maps the sounds in a certain language so that they can be expressed in other scripts and languages, via a phonetic based string layout. The system works by first taking a word, character, or phrase from a target language and giving it a specific IPA designation along with certain Unicode IDs. From here, one can find the corresponding layout in the target language. The patent also protects the layout of the keyboard used for this system.

Essentially, the process can be simplified as a piece of information being transferred from the source language to Unicode then the designated Phonetic String. This string is fed into a phonetic mapping engine which has a fixed scheme that it uses to produce results concisely and reliably. The system also has a series of three large processors that first work together to double check the work that works with different information. For example, Computer 110 works primarily on the backend of this verifying system. Ultimately, this is intended as a reusable program on many different platforms including stationary and mobile for many different environments. It is also applicable to word processing programs. Nevertheless, this is only for text; images are a different matter.

Introducing Semantics

Looking beyond the pronunciation of words, their semantical sides also have weight. The logic naturally embedded into language can be taken advantage of to organize such a large inventory of words. Several natural language processing systems aim to loom for specific keywords or types of keywords that may give hints to the literary subject and predicates of sentences. By identifying these attributes, these programs can turn these hints into a format that is more easily readable by search engine and more robotic operators. Ultimately, a lexer and parser will need to be used to deal with this new fundamental type of data as it provides a more streamlined methodology of dealing with lots of data, a need at the core of Machine Learning. A simplified model that shows the web between some foods is shown below in Figure 3. A semantical tree uses the same approach but with hundreds of thousands of words; this serves as a microcosm to help illustrate a small part of the web.

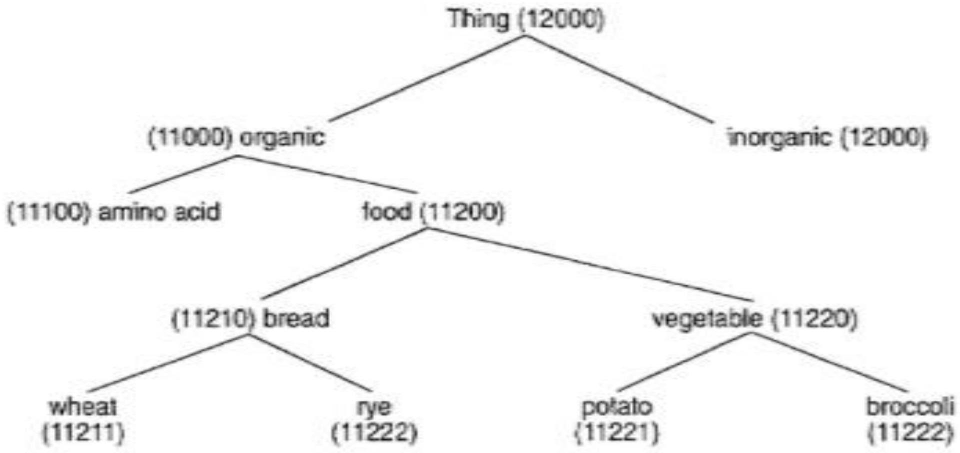


Figure 2. Semantic Dichotomy of Words (Mataic, n.d.)

Semantics in Technology

In a way, there is an ever-growing shift away from translating words by their rough meaning and towards one-to-one translation. Part of this movement includes the new deep learning algorithm developed by Quoc Le and his team at Google (Mataic, n.d.). This part of deep learning is referred to as neural machine translation. This is a series of several layers of processors working together in tandem. The fact that translating language requires deeper cognitive abilities is a debated topic. This was proven due to the old mindset of splicing the sentence which simply did not work sometimes. By using vectors to connect related words and ideas however, they were able to consistently remove most translation errors. This hints at further growth in this field in the nearby future.

Machine Learning for Images

In addition to reading text, reading images is also an important part of this project because it is the first step of getting the shorthand from an image into a format where functions can be performed to get it into English. This can utilize one of two methods (or both): blocking and weighted matrices. Blocking is a method in image recognition that uses distinguishing characteristics to split the image into squares that the algorithm can tackle one by one (Mohamed & Rohm-Ensing, n.d.). From this point, these squares that make up the image can be split up into rows and columns in which certain areas have more weight than others and can be represented with vectors (US8200475B2 - Phonetic-Based Text Input Method - Google Patents, n.d.)

Weighted Matrices

The two main steps of blocking and the weight matrix can be found in many image recognition studies, however, one specific study focused on using these techniques to read Gregg Shorthand. A different method that focused more on thickness was meanwhile used to read Pittman Shorthand (Automatic Recognition and Transcription of Pitman's Handwritten Shorthand—An Approach to Shortforms - ScienceDirect, n.d.). However, to measure the success of the Gregg Shorthand project, the engineers behind this project decided to use two artificial networks both offline and online and compare them in a variety of elements such as computing time, computing power needed, and accuracy (Rajasekaran, 2014). While the online and offline processes differed in the location of the computations (on-server or off-server), their computations were similar.

However, the core weighted matrix system categorized every pixel in a 32 x 32 area into either black or white (Rajasekaran, 2014). 32 x 32 was chosen for its reliability yet is small enough to efficiently work with. A model for this matrix would have to be prepped but after its filtering, softening, sharpening, or embossing, the output would resemble the following.

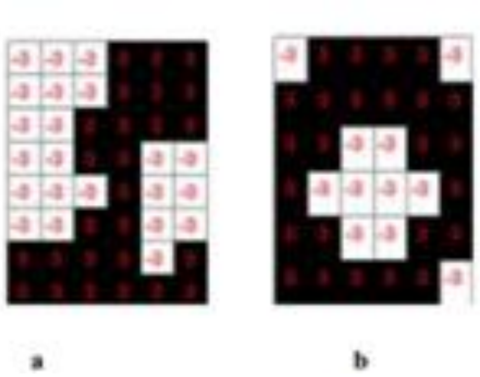


Figure 3. Weighted Matrices for Gregg Shorthand

Expanding on Weighted Matrices

Another study titled, “Automatic recognition and transcription of Pitman's handwritten shorthand—An approach to shortforms,” uses a similar matrix to store information about the image but used more values than just 1s and 0s (Automatic Recognition and Transcription of Pitman’s Handwritten Shorthand—An Approach to Shortforms - ScienceDirect, n.d.). The study starts off by listing a series of unofficial yet extremely helpful and efficient changes to Pitman shorthand including new abbreviations and the changes of certain vowel clusters. To track these changes and to measure whether these changes are significant, they trained an algorithm to read the shorthand and evaluate its efficiency to write and legibility. Examples of their matrix diagrams or processing is shown below.

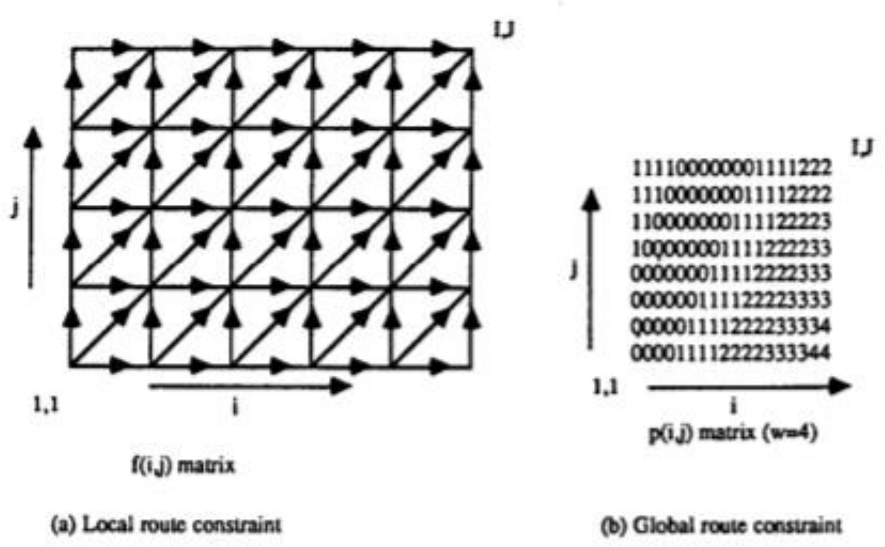


Figure 4. Weighted and Scaled Matrices

To compare this to the Gregg Shorthand example, they took a similar approach to many matrix-based vector analysis procedures as other Handwritten Shorthand Recognition systems. However, the caveat is that in this system it uses a graded scale to gauge it more in detail than just zeros and ones. However, this will lead to dealing with heuristics and thus the workload is heavier on the Machine Learning side.

Monitoring the Direction of Writing with Polar Functions

Another article started by addressing the fact that currently shorthands are important and relevant due to their usage in digital data entry displays. Pitman's wpm rate makes it the ideal tool to use for the job (Segmentation and Recognition of Phonetic Features in Handwritten Pitman Shorthand - ScienceDirect, n.d.). First, the article works on analyzing the patterns within the strokes of Pitman Shorthand. It marks the change of direction and the order to check how many times the direction of writing changes. From there, the samples must be segmented so that the flow of different characters combined with each other can be realized and to prep for the Box Model. This Box Model is a helpful tool for AI to break up sensing objects or features in visual data. From here the computer represents the strokes as a series of polar functions and applies the summative vectors and weighted matrices approach standard in Machine Learning. Here is a diagram of the preliminary portion however.

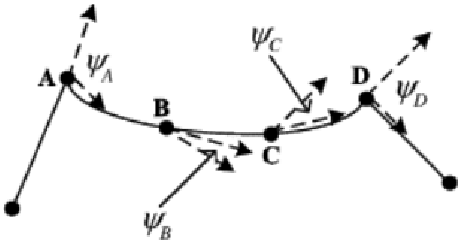


Figure 5. Directional Movements in Pitman

Conclusion

By using these various facets of Machine Learning, Optical Character Recognition (OCR) and related work pertaining to the digital reading of shorthands has been made more efficient and accurate thanks to numerous researchers. This includes laying the foundation of the concept of maximizing the information density by exploiting the difference in Information Rate versus Speech Rate, then applying it to Unicode and pairing it with IPA in the digital world. From here, teams have experimented with ways of incorporating semantics to better sort through data. Then, image processing must be trained with blocking and matrices to feed this information into the text system. Only then can a cohesive tool be built for reading an efficient shorthand developed last year.

Future Applications

Once this project can effectively translate information from writing to digital text, this project can be embedded into an app where it is more easily accessible for the public. The focus of this application would be the integration and usage with other applications on touch-screen devices. One possible idea is that the user would use the interface in place of the traditional keyboard on smartphones.

Shorthands also have a large prevalence in the medical industry. Despite the many guidelines for the use of abbreviations and shorthand in the medical industry, it can still be difficult to standardize conventions at times (Politis, n.d.). This program could help bridge the gaps by either promoting a universal medical shorthand or translating between existing solutions.

Therefore, software can close communication gaps in industries and is also capable of advocating for universal standards.

Thesis

Living in the Information Age, society must make its numerous inventions and tools computer-compatible, including shorthands. One of the first shorthands created was Pitman Shorthand in the 1800s to keep up with the demand of denoting information on paper (Automatic Recognition and Transcription of Pitman's Handwritten Shorthand—An Approach to Shortforms - ScienceDirect, n.d.). But because of technological advancements however, paper transcription became obsolete thus no longer supporting Shorthands as an efficient medium of communication.

To bring shorthands into the 21st Century includes developing systems that can quickly parse through the writing autonomously. At first, researchers focused on existing writing systems. They explored how to identify characters from conventional orthography from languages such as English and Arabic (Mohamed & Rohm-Ensing, n.d.), but others soon turned to shorthands as they have faster input speeds, measured typically in words per minute (Automatic Recognition and Transcription of Pitman's Handwritten Shorthand—An Approach to

Shortforms - ScienceDirect, n.d.). In the case of a group researching Gregg Shorthand (widely regarded as one of the world’s most popular shorthands), they used an artificial neural network to translate handwritten Gregg Shorthand to digital text using Machine Learning techniques such as blocking quickly and accurately (Rajasekaran, 2014). Likewise, a group researching Pitman Shorthand focused on the changes in direction and used weighted matrices like the Gregg Shorthand team to convert the information to digital text (Automatic Recognition and Transcription of Pitman’s Handwritten Shorthand—An Approach to Shortforms - ScienceDirect, n.d.). Nevertheless, the common thread is groups are focused on digitalizing the writing for speed.

Therefore, building a model around the new shorthand developed last year with Omer Alptekin to outdo Gregg Shorthand in terms of efficiency was built upon the work of researchers in this field by applying similar techniques to a newly optimized Shorthand (Developing a New Shorthand - YouTube, n.d.). The new Digital Shorthand Key, however, is shown to have higher information density (1.52 pixels/stroke) and take less time to write than Gregg Shorthand (1.34 pixels/stroke) because of its fewer pixels and strokes. To gauge information density, the project coined the term Speed Score. This was a unique metric developed for the usage of this project last year which involved taking the average RGB value of a written sample and dividing it by the sample’s stroke count. On the RGB scale, 0 was black and 255 was white. The reasoning behind the metric was to reward less writing, leading to a higher RGB score, and fewer strokes, leading to a smaller denominator, with a higher Speed Score (Developing a New Shorthand - YouTube, n.d.). The Speed Score was the determining characteristic in choosing E3, or experimental group 3, to serve as the Digital Shorthand Key, a critical part of the final product: an open-source

Machine Learning model with a corresponding Graphical User Interface to enable users to use an automated system to save time writing down their information via the new shorthand.

Methods

While there are several technologies capable of creating this final product, Tensorflow, Jupyter Notebooks, and Google Colab are popular among this subset of Machine Learning as they are designed to support image classification using a variety of existing Python libraries such as Numpy and Pandas (G D, 2018). Among these three, using Tensorflow directly from the command line along with code from a Github repository created by Steven Dufresne to identify playing cards seemed like the best option as it allowed for maximum customization (Dufrense, 2018).

To set up the virtual environment needed for this approach, Anaconda, CUDA, and cuDNN were installed to take advantage of the NVDIA Graphics Card built into the computer; These steps are exclusive to those wishing to pursue a model powered by Tensorflow GPU, as was used in this project.

After downloading the software, new directories were made, and their PYTHONPATHs were linked via the System Path Manager. In these new directories, a model from the Official Tensorflow Model Zoo Repository was copied to the models directory. The model chosen for this project was the Faster-RCNN-Inception-V2-COCO model as it was shown to only take about 3 hours to train to an optimized level. It was also a good choice as Recursive Neural Networks are best if used for tasks like this, recognizing lines and curves, because they can use backwards propagation, going back to earlier layers, to produce a more accurate final dichotomy (Shi, n.d.).

The next two steps in more detail can be found under steps 2d and 2e in the card detection program: set up Tensorflow with Anaconda in command prompt along with supplemental libraries and configure PYTHONPATH (Dufrense, 2018). This incorporates Tensorflow into the virtual environment. Then, since Tensorflow uses pb2.py files, a protoc command compiles existing files into a format the Faster RCNN model can use.

Once Tensorflow is set up, the first step of customizing the model is acquiring the data on which the model will be trained. For writing samples of the shorthand, one of Aesop's famous texts "The North Wind and the Sun" was utilized as it encompasses most English phonemes. The whole text was written twice, and each writing was split into two halves. The first half of the second repetition was appended to the test directory and the rest was placed in the train directory.

Next, the pictures were annotated with LabelImg to create corresponding xml files. These files could then be turned into csv files Tensorflow can reading via the xml_to_csv.py executable. Then in the generate_tfrecord.py file and labelmap.py in the training directory, the number of classes was changed to 12 and the corresponding named were paired with the classes too. The input paths and label_map_paths were also changed to match with the train and test directory. Finally, train.py was ran with the corresponding options and the program began to move through the steps.

Each step can be thought to be a different iteration of the same underlying model. In each step, the model has access to the same control characters in the train directory. The characters that the model is exposed to also serve as a highly accurate representation of common English phonology with balanced frequencies for each phoneme since the shorthand is simply a translation of Aesop's original story.

Many of the results from the experiment are automatically recorded with Tensorflow's Tensorboard. Tensorboard has a variety of loss or inaccuracy graphs, but for the purpose of this project, priority will be given to the Total Loss graph. In addition, a One Proportion Z-Test was used to evaluate the credibility of the model by comparing observed success rate to a lucky guess from the 12 Digital Shorthand Key characters.

Results

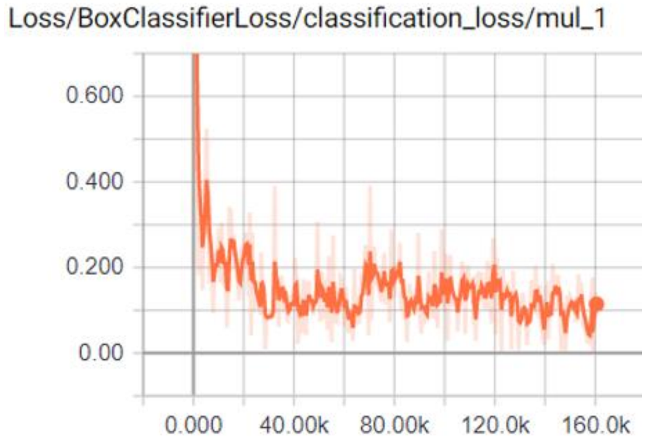


Figure 6. Total Loss from 0 Steps to 160,000 Steps

This is a graph of the progression of Loss (inaccuracy) on the y-axis over time measured in training steps on the x-axis.

The whole experiment or training of the model ran for about 3 hours and 160,000 steps. In the end, the model reached a loss or inaccuracy of about 11.37%. This means the model can accurately read one of the 12 characters from the Digital Shorthand Key.

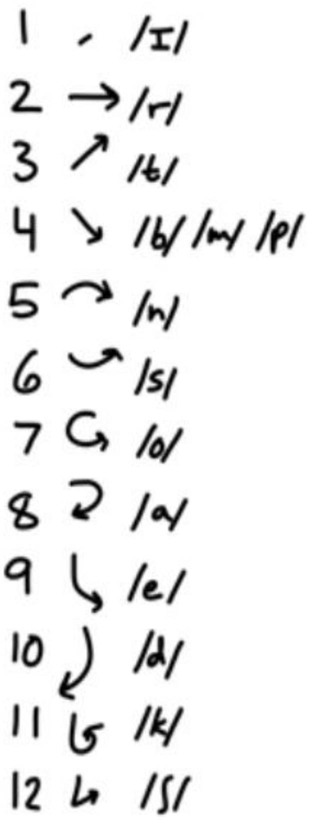


Figure 7. Digital Shorthand Key Characters

The following is a list of all 12 Digital Shorthand Key characters and their corresponding phoneme and symbol.

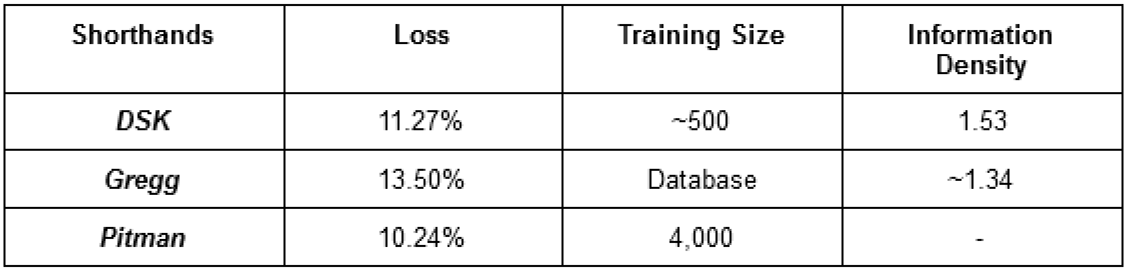
To check the validity of the model, and to confirm these results were not the product of random chance, a One Proportion Z-Test can be used to find a P-value (One Sample Test of Proportions, 2016). Please refer to the Appendix for more information.

The data yielded a z-score of ~39.08 standard deviations from the mean. This translates to a $P \leq 0.00001$, which shows the recognition accuracy of the model is significant ($P \leq 0.05$). It is most probable these results are not due to random chance, and thus the Machine Learning algorithm is effective when applied to the shorthand.

Furthermore, when compared to similar systems made for Gregg Shorthand and Pitman Shorthand, the Digital Shorthand Key meets the benchmark for accuracy despite only being trained for about 500 samples.

Discussion

Since there is evidence to support the notion the Machine Learning algorithm is largely successful and the shorthand itself is shown to a significant advantage when measuring by Speed Score, it can be asserted the Digital Shorthand Key serves as an optimal method for faster transcription. The p-values significantly less than 0.05 also reinforce the ideas that the Machine Learning model is successful almost 90% of the time and that the chance of Gregg Shorthand outdoing the Digital Shorthand Key in terms of Speed Score is less than 1%. Hence, the Digital Shorthand Key is a reliable and efficient medium for quickly transcribing text that embodies an image recognition and English word mapping elements.



Shorthands	Loss	Training Size	Information Density
DSK	11.27%	~500	1.53
Gregg	13.50%	Database	~1.34
Pitman	10.24%	4,000	-

Table 2. Comparison of Machine Learning Models Across Shorthands

This Table compares the Digital Shorthand Key, Gregg, and Pitman in terms of Loss, Training Size, and Information Density. Higher Information Density indicates a more efficient Shorthand and more efficient Algorithm as well.

However, the real strength of using Machine Learning for the Digital Shorthand Key lies in the Information Density of the Shorthand, as determined by research done last year (Developing a New Shorthand - YouTube, n.d.). To support the notion that the Digital Shorthand Key is also more Informationally Dense than Gregg Shorthand as part of the work last year, a Student’s T-Test was conducted to measure the probability that the ranges of the Information

Densities of different experimental groups could overlap. Speed Score was calculated by dividing the unused pixels as part of the RGB average of a sample by the number of strokes.

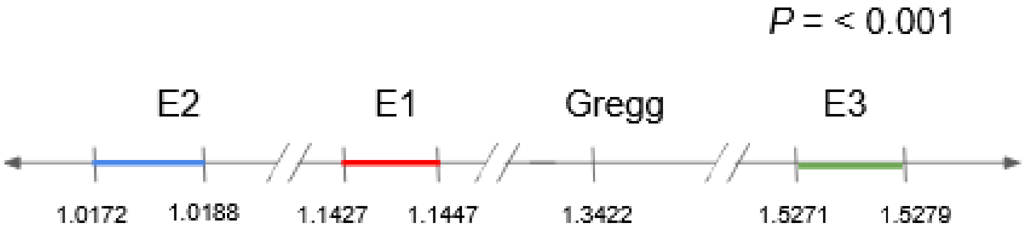


Figure 8. Ranges of Information Density Between Experimental Groups and Gregg

This number line shows where the Information Density of each group (±3 standard deviations) lies.

In this statistical test done last year, each experimental group contains 30 samples except for Gregg Shorthand, since Gregg Shorthand samples were obtained from an Online Translator (Šarman, n.d.). This translates to a $P \leq 0.0001$, which shows that likelihood of Experimental Group 3, now known as the Digital Shorthand Key, being more efficient than Gregg Shorthand is significant ($P \leq 0.05$) (Developing a New Shorthand - YouTube, n.d.). It is most probable that these results are not due to random chance, and thus the notion the Shorthand is more efficient than Gregg shorthand is supported.

Nevertheless, specialized equipment, such as Steno Machines, also serve as a popular alternative for those looking to write at high speeds at the price of extra equipment (MacMillan, 2016). However, the Digital Shorthand Key aims to provide an equivalent free for all and available at a moment's notice.

The Digital Shorthand Key also proves itself to be an improvement upon a Pitman Shorthand recognition system. The study focused on interpreting Pitman shorthand achieved a loss of about 10%, proving to be a formidable competitor, but the authors also conceded there were a reasonable number of common errors among their shorthand experts (Ma et al., 2008). From analyzing the mistakes, the researchers also suggested a variety of amendments to the shorthand to make it more user friendly. While Machine Learning algorithms can be applied to any shorthand, it takes a new more efficient shorthand to surpass its competitors.

This project itself is an application of the observation made in a linguistics study. The study published on ScienceMag conducted an experiment in which they compared languages in terms of information density and determined that there was indeed sizeable variance across languages (Coupe et al., 2019).

Likewise, studies and articles on Unicode also detail how the repetition in language can be leveraged in text file compression (Studený, n.d.). In terms of Unicode, it creates a HashMap-like dictionary of shorter terms to represent strings, longer terms.

The model was trained exclusively using common English words in famous linguistic-used stories and the most common words in the English lexicon. When the model is confronted with new words or slang it may not recognize, it may be comfortable recognizing characters that it has gotten used to and assume incorrect meanings. In this scenario, the accuracy of the model would be an overestimate.

Likewise, the loss of the model would fluctuate greatly even among a smaller range of ~5,000 steps. This allows from interpretations of the total loss to be anywhere from 0.5 to 0.13. In this regard, the loss chosen to be reported, 0.113, would be an underestimation of the model's abilities and success.

Similarly, errors within the application will always persist too. Ambiguity in language found by computers is one of the biggest problems present whenever computers meet language and at the core of Natural Language Processing (Nadkarni et al., 2011). A future avenue of study would be to reduce the little amount of ambiguity this system is bound to face. This can be done via variety of methods, name some low-level Natural Language Processing tasks such as Part-of-Speech assignation.

The ontology-based parser from Busch serves an excellent example of an existing patent using this step in Natural Language Processing methodology (Busch et al., 2001). The invention takes input, translates it phonetically, and tracks the part of speech to create a more fluid parser for typing applications. This project builds upon this work by using this element of the methodology to solve another text input problem, albeit not via text but rather shorthand.

To understand how this solution may impact this product, look at the word identification process once again. This method of going back from the Digital Shorthand Key via an organized dichotomy is like how researchers at Google used a new Artificial Intelligence to dramatically increase the translation success of words and phrases (Mataic, 2016). In Google program, they split all the words up into trees with different subbranches. For example, you would find the word “dog” on the “pets” branch on the “animals” branch on the “living” branch. In the case of the Digital Shorthand Key, however, the twelve characters in the Digital Shorthand Key would see branches less like the example on the left, Google’s semantic tree, and more like the example on the right, representative of this project and shows the possible interpretations for the letters t and o in the word.

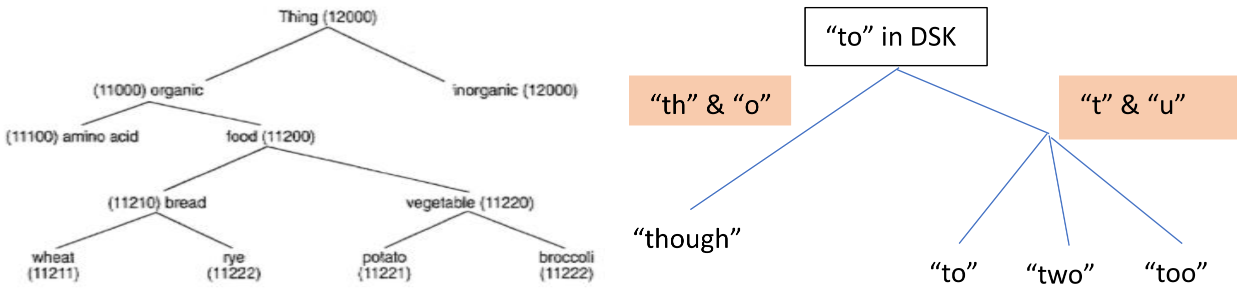


Figure 9. and 10. Semantic Trees for Words

The following two trees show how a team from Google classifies words (left) and how the Digital Shorthand Key identifies words.

Since "to," "two," "too," and "though" are going to typically found in different parts of sentences, some of these context clues embedded naturally into language may also help solve this current error. These are the same context clues people apply when distinguishing homophones verbally. As of right now however, this is a minor problem as the algorithm is designed to allow the user to choose from multiple possible interpretations.

In all, the system provides a quick and easy way to communicate information through this model majority of the time. It provides a quantitative metric of the effectiveness of the trained model and a new medium for faster digital transcription for everyone in all environments. The digital shorthand system can help people save time writing on touchscreen devices such as iPads, write without the traditional keyboard (easier for the visually impaired), and standardize or facilitate shorthands in the medical industry.

References

Busch, J., Lin, A., Graydon, J., & Caudill, M. (2001). Ontology-based parser for natural language processing—Google Patents. Retrieved November 24, 2020, from https://patents.google.com/patent/US7027974B1/en

Coupe, C., Mi Oh, Y., Dediu, D., & Pellegrino, F. (2019). Different languages, similar encoding efficiency: Comparable information rates across the human communicative niche | Science Advances. Retrieved November 24, 2020, from https://advances.sciencemag.org/content/5/9/eaaw2594

Developing a New Shorthand—YouTube. (n.d.). Retrieved March 6, 2021, from https://www.youtube.com/watch?v=yxnAJrMxrAI&feature=youtu.be

Dufrense, S. (2018, May 23). Using TensorFlow To Recognize Your Own Objects | Hackaday. https://hackaday.com/2018/05/23/using-tensorflow-to-recognize-your-own-objects/

G D, D. (2018, December 14). Image Recognition With Colab and Jupyter with wget—YouTube. https://www.youtube.com/watch?v=NOg18lr6ZmY

Kotipalli, K. (n.d.). Phonetic-based text input method. Retrieved November 24, 2020, from https://patents.google.com/patent/US8200475B2/en

Leedham, C. G., & Downton, A. C. (n.d.). Automatic recognition and transcription of Pitman's handwritten shorthand—An approach to shortforms—ScienceDirect. Retrieved November 24, 2020, from https://www.sciencedirect.com/science/article/abs/pii/0031320387900082

Ma, Y., Leedham, G., Higgins, C., & Htwe, S. M. (2008). Segmentation and recognition of phonetic features in handwritten Pitman shorthand. *Pattern Recognition*, *41*(4), 1280–1294. https://doi.org/10.1016/j.patcog.2007.10.014

MacMillan, T. (2016). How to Type 360 Words a Minute. Retrieved December 22, 2020, from https://nymag.com/speed/2016/12/how-to-type-360-words-a-minute.html

Mataic, C. (2016). Google's new translation software is powered by brainlike artificial intelligence | Science | AAAS. Retrieved November 24, 2020, from https://www.sciencemag.org/news/2016/09/google-s-new-translation-software-powered-brainlike-artificial-intelligence

Mohamed, A., & Rohm-Ensing, E. (n.d.). *English-Arabic Handwritten Character Recognition using Convolutional Neural Networks*. 7.

Nadkarni, P., Machado, L., & Chapman, W. (2011). Natural language processing: An introduction | Journal of the American Medical Informatics Association | Oxford Academic. Retrieved November 24, 2020, from https://academic.oup.com/jamia/article/18/5/544/829676

One Sample Test of Proportions. (2016). Retrieved January 31, 2021, from https://sphweb.bumc.bu.edu/otlt/MPH-Modules/BS/SAS/SAS6-CategoricalData/SAS6-CategoricalData2.html

Politis, J. (n.d.). Overview of shorthand medical glossary (OMG) study—Politis—2015—Internal Medicine Journal—Wiley Online Library. Retrieved December 5, 2020, from https://onlinelibrary.wiley.com/doi/abs/10.1111/imj.12668

R. Rajasekaran, Dr. K. R. (2014). Statistical review of Online/Offline Gregg Shorthand Recognition using CANN and BP - A Comparative analysis. *IJAIST*, *24*(24), 13.

Šarman, S. J. (n.d.). steno: Gregg. Retrieved January 31, 2021, from https://steno.tu-clausthal.de/Gregg.php

Shi, Y. (n.d.). TensorFlow 1 Detection Model Zoo. Retrieved January 31, 2021, from https://github.com/tensorflow/models/blob/master/research/object_detection/g3doc/tf1_detection_zoo.md

Studený, P., & Holeček, O. (n.d.). Fast Compression Algorithm for UNICODE Text. Retrieved November 24, 2020, from http://unicode.org/notes/tn31/

Touchscreen typing speeds close in on keyboard rates—BBC News. (2019, October 4). https://www.bbc.com/news/technology-49933204

Appendix

Special thanks goes to the following people for help with this project:

- Dr. Kevin Crowthers
- Mrs. Angela Taricco
- Mr. William Ellis
- Andrew Yang
- Rumaisa Abdulhai

Altogether, the limitations can be summarized as:

- The current lack of an app
- Compatibility with only the English Language
- The ambiguity with homonyms
- Educating people how to use the shorthand
- Working with proper nouns

These problems can be addressed by:

- Building an interactive app that features a frontend to work with developed backend
- Analyzing the frequency patterns in other languages
- Please see Figure 9 and 10 in the Discussion
- The app may include a practice/training mode to help people practice
- The repetition of certain characters can be used to denote specific characters as needed

Analysis (continued)

The following discusses the usage of the One-Proportional Z-Test, featured via the equation below.

$$z = \frac{\hat{p} - p_0}{\sqrt{\frac{p_0(1 - p_0)}{n}}}$$

$$\hat{p} = Observed\ Population$$

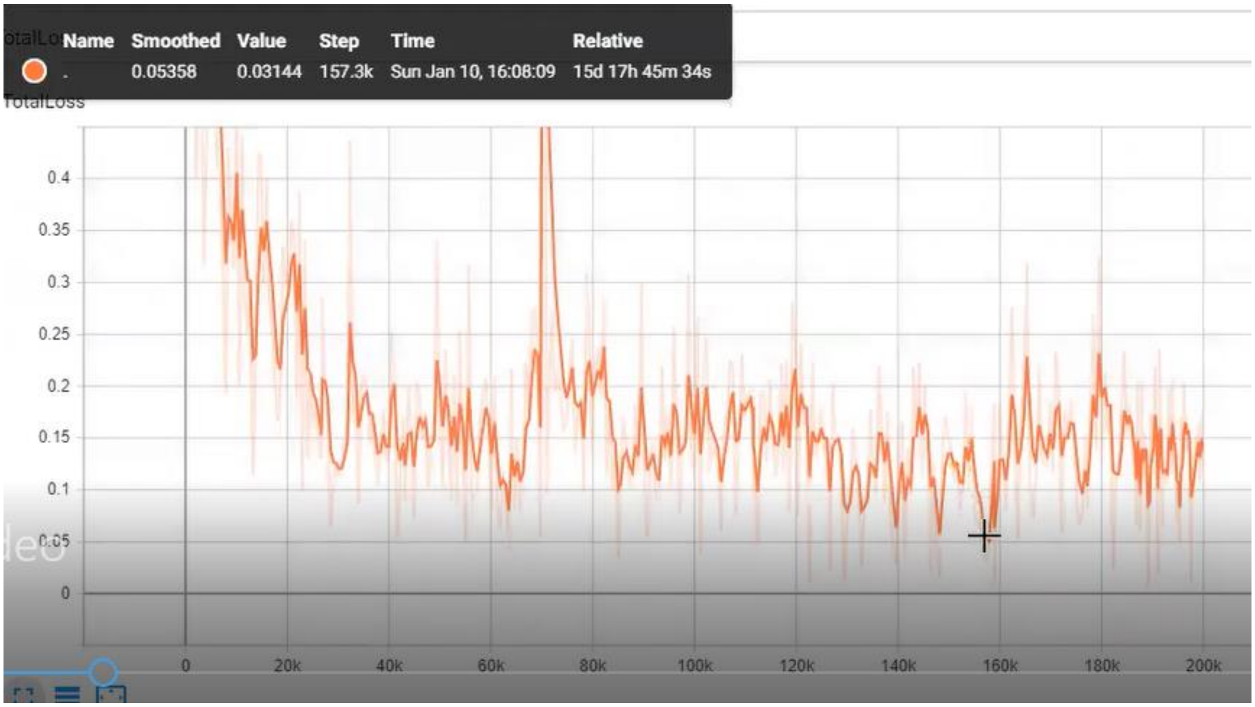
$$p_0 = Null\ Hypothesized\ Value$$

$$n = Sample\ Size$$

$$z = Z\ Score$$

In this project, the observed population was $\frac{154}{174}$ because there were 174 characters in the test directory of which 20, ~11% of 174, were incorrectly identified. Conducting this test on the data yielded a z-score of ~39.08 standard deviations from the mean. This translates to a $P \leq 0.00001$, which shows the recognition accuracy of the model is significant ($P \leq 0.05$).

Photos



Name	Smoothed	Value	Step	Time	Relative
.	0.05358	0.03144	157.3k	Sun Jan 10, 16:08:09	15d 17h 45m 34s

Figure 6. Interactive TensorBoard Loss Graph

This graph shows the efficacy in training the model over time, represented in steps on the x-axis, and loss, represented in loss as a decimal on the y-axis.

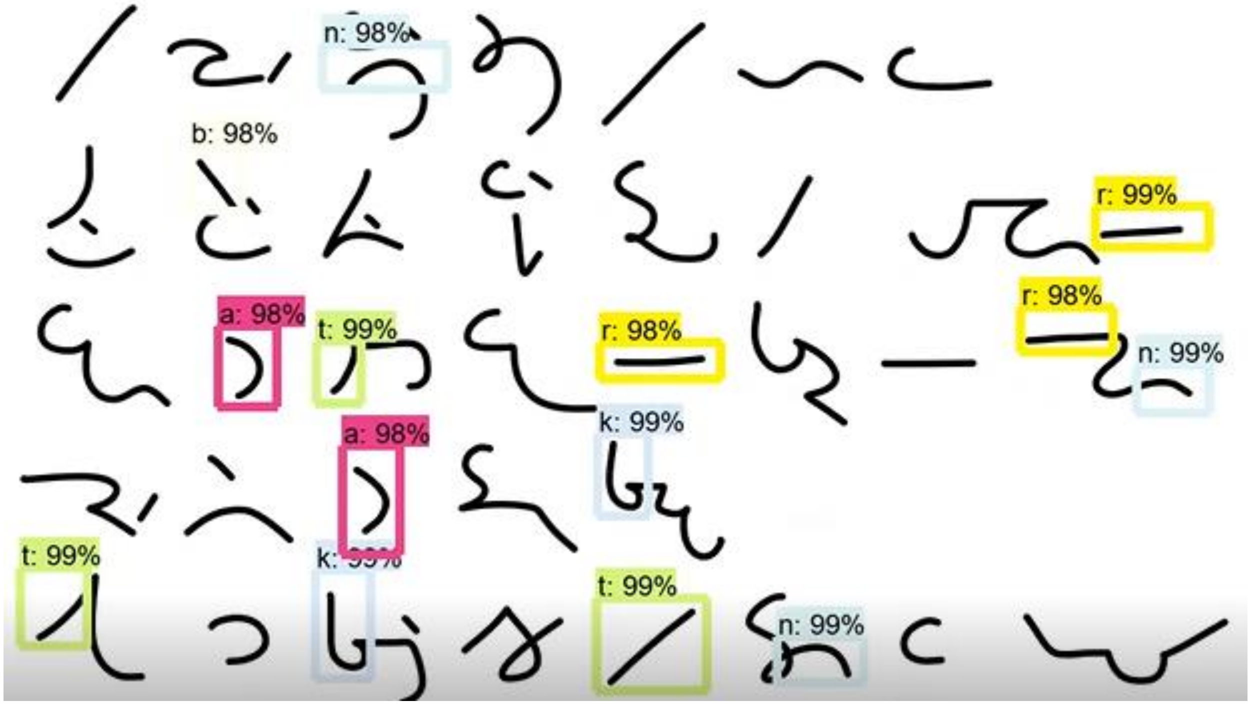
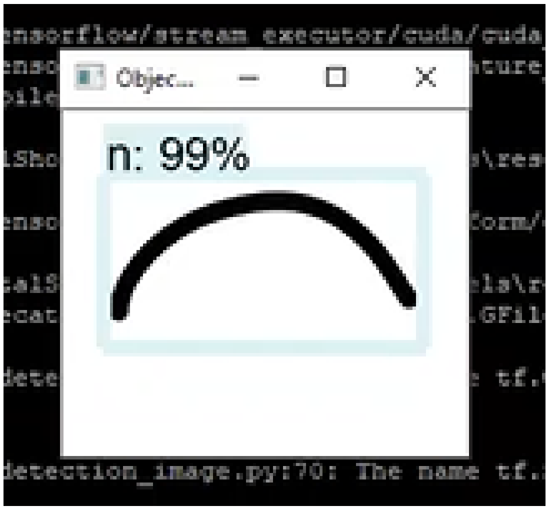


Figure 7. Output of Identifying Hundreds of Characters

This shows the famous linguistic text “The North Wind and The Sun” referenced earlier but with annotations by Tensorflow with the help of Python. In this sample, only select characters are annotated as to not overwhelm the program.



```
ensorflow/stream executor/cuda/cuda
enso                                   ture
pile
lSho                                   s\res
enso                                   form/
talS                                   ls\r
ecat                                   .GFil
dete                                   : tf.
detection_image.py:70: The name tf.
```

Figure 8. Output of Identifying a Singular Character

When a few characters or a singular character is presented to the program, the program can confidently identify the character. In this sample, the character “n” is shown.

Decision Matrix

Shorthand	**Info Density (10)**	**Loss (7)**	**Training Size (6)**	**Total**
DSK	10	8	8	204
Gregg	8	6	2	134
Pitman	-	9	5	183

Figure 9. Decision Matrix to Compare Machine Learning Algorithms for Different Shorthands

This table gives weight to three categories and summates their parts to get a sum representative of the total performance of the system. Information density is directly related to Speed Score and a higher score is given to shorthands with smaller losses, or higher accuracy. Higher scores are given to shorthands with a smaller training size because of the potential they have to grow with the growth of data.

Project Notes

The following includes my project notes, which draw from a variety of sources to constitute my research behind this project.

Knowledge Gaps

This list provides a brief overview of the major knowledge gaps for this project, how they were resolved and where to find the information.

Knowledge Gap	Resolved By	Information is located	Date resolved
Research ML Softwares that could prove to be useful for my project	Creating a plan for the next two months and all the material I will need for studying	- MIT OpenCourseware for Linear Algebra - 3Blue1Brown for Essence of Calculus - Probability by MIT on EdX - 3 Playlists for - Math - Tensorflow - Deep Learning - Unacademy course ML	9/30

Research how to download and connect the software myself	Finding some tutorials online (on youtube for example) of importing Tensorflow and its respective libraries	Youtube Tutorial	9/27
Lexers and Parsers	Meeting with Mrs. Taricco Watching YouTube tutorials Reading articles about the application and purpose about mainly lexers	Lexers and Parsers	10/14

Literature Search Parameters

These searches were performed between (Start Date of reading) and XX/XX/2019.

List of keywords and databases used during this project.

Database/search engine	Keywords	Summary of search
Science Mag	Handwritten Recognition AI Translation Information Density Efficiency + Language Natural Language Processing	Google's new translation software is powered by brainlike artificial intelligence Automatic recognition and transcription of Pitman's handwritten shorthand—An approach to shortforms (I had to consult multiple sources for this article across databases)
Google Scholar	Patents Unicode Input	English-Arabic Handwritten Character Recognition using Convolutional Neural Networks Method for computer-assisted translation

	Natural Language Processing Shorthand	Phonetic-based text input method
Google	Character recognition Machine Learning Gregg Pitman Shorthand	Statistical review of Online/Offline Gregg Shorthand Recognition using CANN and BP - A Comparative analysis Segmentation and recognition of phonetic features in handwritten Pitman shorthand

Article #1 Notes: New Translation Software

Article notes should be on separate sheets

Source Title	Google's new translation software is powered by brainlike artificial intelligence https://patents.google.com/patent/US8200475B2/en.
Source citation (APA Format)	icSep. 27, C., 2016, & Pm, 2:45. (2016, September 27). *Google's new translation software is powered by brainlike artificial intelligence*. Science | AAAS. https://www.sciencemag.org/news/2016/09/google-s-new-translation-software-powered-brainlike-artificial-intelligence
Original URL	https://www.sciencemag.org/news/2016/09/google-s-new-translation-software-powered-brainlike-artificial-intelligence
Source type	Digital News
Keywords	Technology, AI, Translation
Summary of key points (include methodology)	A team from Google Mountain View, California led by Quoc Le used a new deep learning algorithm to reduce errors by finding the nuances in languages not explicitly stated. By using these sensitive systems, they were able to find the small, illusive details of language. This article is also a summary on how translation is being made easier through a process called vector processing. Vectors are these relations between

	words that cannot be stated objectively. By asserting the surmounting task of locating and dealing with each minor linguistic anomaly, data scientists can make difficult tasks like translation easier.
Research Question/Problem/ Need	How can digital translators be made more accurate?
Important Figures	It was 58% more accurate at translating English into Chinese, and 87% more accurate at translating English into Spanish Vectors: 2.5 billion sentence pairs for English and French; 500 million for English and Chinese **reduces translation errors by up to 87%**. "This … demonstrates like never before the power of neural machine translation," says Yoshua Bengio The new method, reported today on the preprint server arXiv, uses a total of 16 processors to first transform words into a value known as a vector

	A lot of the inspiration for this paper came from [the fields of] speech and computer vision," says Thang Luong, a graduate at Stanford University in Palo Alto, California
Notes	There is an ever-growing shift away from the rough idea and towards one-to-one translation. Part of this movement includes the new deep learning algorithm developed by Quoc Le and his team at Google. This part of deep learning is referred to as neural machine translation. This is a series of several layers of processors working together in tandem. The fact that translating language requires deeper cognitive abilities is a debated topic. This was proven due to the old mindset of splicing the sentence which simply did not work sometimes. By using vectors to connect related words and ideas however, they were able to consistently remove the majority of translation errors. This hints at further growth in this field in the nearby future.
Cited references to follow up on	https://arxiv.org/pdf/1609.08144v1.pdf https://www.sciencemag.org/news/2016/01/huge-leap-forward-computer-mimics-human-brain-beats-professional-game-go
Follow up Questions	How can we track vectors?

	What are some semantic ideas and details that were unable to be picked up with the DL system but not my foresight? Can this be combined with sentence splicing in any way?

Article #2 Notes: Method for computer-assisted translation

Article notes should be on separate sheets

Source Title	Method for computer-assisted translation
Source citation (APA Format) (Mercier, 2001)	er, P. (2001). *US20030105621A1—Method for computer-assisted translation—Google Patents.* https://patents.google.com/patent/US20030105621A1/en
Original URL	https://patents.google.com/patent/US20030105621A1/en
Source type	Patent
Keywords	Translation, Remote translation, Multiple Input, Translation sequence proposals
Summary of key points (include methodology)	The patent protects a method of integrating several computers and translator servers through a communications server. The benefits of this specific system are that it allows for the target and source language sequence to be backed up and utilized for future use by the database The system also gives simultaneous access to other users and other more remote translation servers that not only enhances efficiency but also is quite secure. The system tries to be as conservative as possible by trying

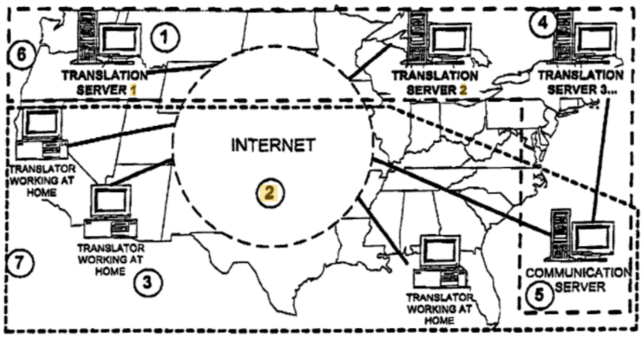
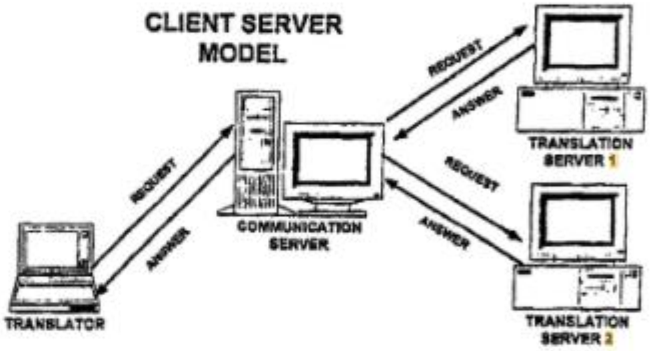
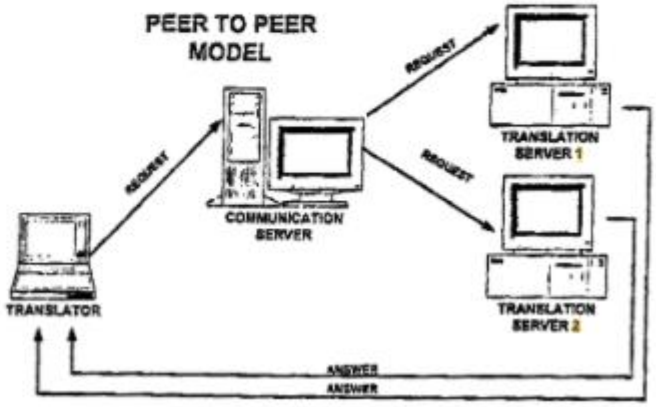
<table>
<tr><td></td><td>to compare the source of any translation request with past queries and comparing similarity percentages</td></tr>
<tr><td>Research Question/Problem/ Need</td><td>Can we quickly and effectively translate using a variety of programs and computers should be easily accessible to each and every computer with this system?</td></tr>
<tr><td>Important Figures</td><td>


FIG. 3
</td></tr>
</table>

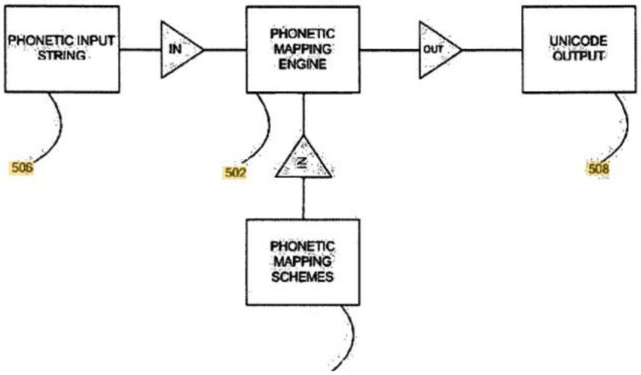
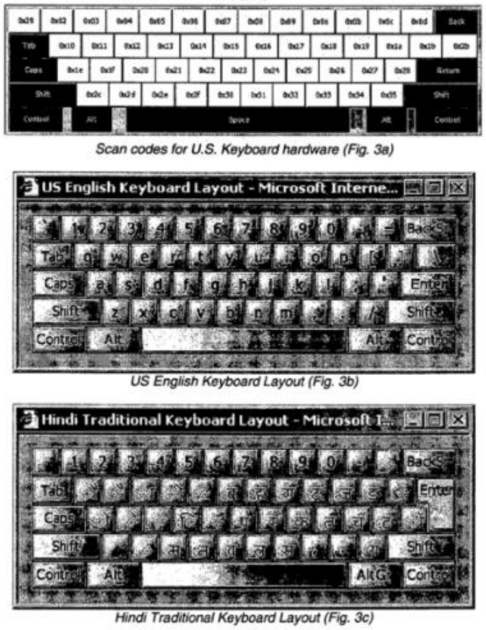
	 CLIENT SERVER MODEL REQUEST ANSWER REQUEST ANSWER REQUEST ANSWER COMMUNICATION SERVER TRANSLATOR TRANSLATION SERVER 1 TRANSLATION SERVER 2 FIG. 8  PEER TO PEER MODEL REQUEST REQUEST REQUEST COMMUNICATION SERVER TRANSLATOR TRANSLATION SERVER 1 TRANSLATION SERVER 2 ANSWER ANSWER FIG. 9
Notes	The following patent is a cross-verification/reinforcement language translating system that works by delegating a variety of tasks in a specific manner across many computers and servers. Connects with two other computers that mainly help with storage and also previous quote and syntax recalling.

Cited references to follow up on	https://patents.google.com/patent/US4706212A/en https://patents.google.com/patent/US5568383A/en https://patents.google.com/patent/US5848386A/en
Follow up Questions	Why did they choose to take the percentage in common over other methods? How often does this process take? How many servers are usually working together at a given time?

Article #3 Notes: Phonetic-based text input method

Article notes should be on separate sheets

Source Title	Phonetic-based text input method
Source citation (APA Format) (Kotipalli, 2012)	Kotipalli, Krishna V. Phonetic-based text input method. United States US8200475B2, filed February 13, 2004, and issued June 12, 2012. https://patents.google.com/patent/US8200475B2/en.
Original URL	https://patents.google.com/patent/US8200475B2/en
Source type	Patent
Keywords	Character encoding, phonetics, text input, keyboard layout, phonetic key
Summary of key points (include methodology)	In order for one to accurately map the sounds in a certain language so that they can be expressed in other scripts and languages, one must use this phonetic based string layout. The system works by first taking a word, character, or phrase from a target language and giving it a specific IPA designation along with certain unicode IDs. From here, one can find the corresponding layout in the target language. The patent also protects the layout of the keyboard used for this system.

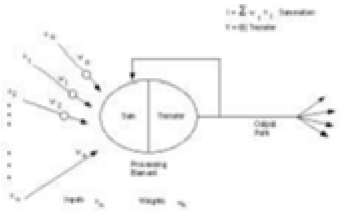
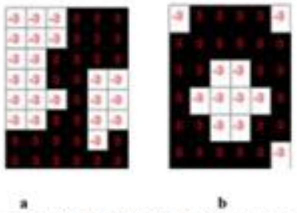
Research Question/Problem/ Need	Can we develop a way of inputting a universal phonetic for sounds in one language to be accurately represented in another?
Important Figures	 PHONETIC INPUT STRING IN PHONETIC MAPPING ENGINE OUT UNICODE OUTPUT 506 502 IN 508 PHONETIC MAPPING SCHEMES FIG. 3  Tab Caps Shift Control Alt Space Back Return US English Keyboard Layout - Microsoft Interne... Hindi Traditional Keyboard Layout - Microsoft I... Enter AltGr Scan codes for U.S. Keyboard hardware (Fig. 3a) US English Keyboard Layout (Fig. 3b) Hindi Traditional Keyboard Layout (Fig. 3c)
Notes	Source Language >> Unicode >> Phonetic String >> Analysis >> Unicode >> Target Language

	The phonetic mapping engine has a fixed scheme that it uses to concisely and reliably produce results. This is intended as a reusable program on many different platforms including stationary and mobile for many different environments. It is also applicable to word processing programs. The system also has a series of three large processors that first work together to double check the work that works with different information for example Computer 110 works primarily on the backend.
Cited references to follow up on	https://patents.google.com/patent/US4731735A/en https://patents.google.com/patent/US5047932A/en https://patents.google.com/patent/US5136504A/en https://patents.google.com/patent/US5243519A/en
Follow up Questions	What steps did they take to streamline the process? At what points did the program think it was most crucial to double check its work? How could they take frequency into account when assigning IDs in their own system before Unicode?

Article #4 Notes: Statistical review of Online/Offline Gregg Shorthand Recognition using CANN and BP - A Comparative analysis

Article notes should be on separate sheets

Source Title	Statistical review of Online/Offline Gregg Shorthand Recognition using CANN and BP - A Comparative analysis
Source citation (APA Format)	R. Rajasekaran, Dr. K. R. (2014). Statistical review of Online/Offline Gregg Shorthand Recognition using CANN and BP - A Comparative analysis. *IJAIST*, *24*(24), 13.
Original URL	https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.672.2819&rep=rep1&type=pdf
Source type	Journal Article
Keywords	Handwriting recognition, DIU, machine learning, Gregg Shorthand
Summary of key points (include methodology)	The engineers behind this project decided to use two artificial networks both offline and online and compare them in a variety of elements such as computing time, computing power needed, and accuracy. While the online and offline processes differed, they both followed this general outline shown in Figure 1A. A weighted matrix system categorized every pixel in a 32 x 32 area into either black or white. 32 x 32 was chosen for its reliability yet is small

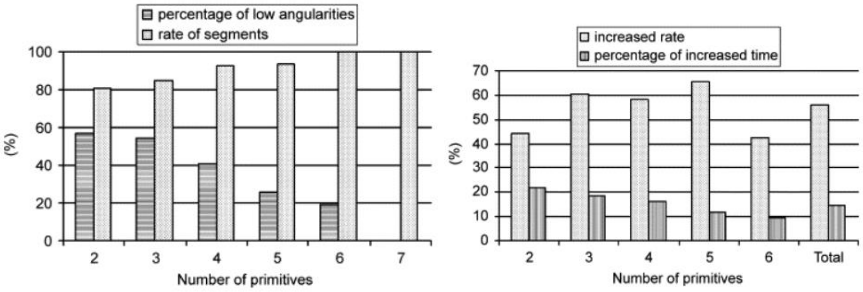
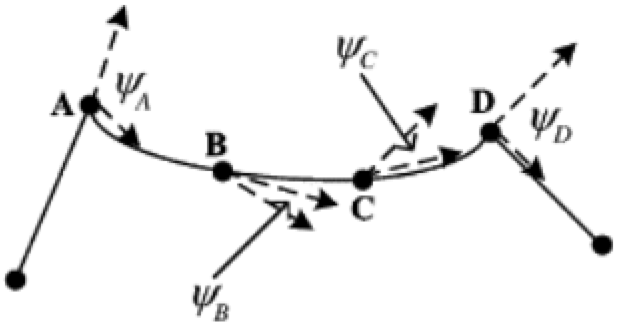
	enough to efficiently work with. A model for this matrix would have to be prepped but after its filtering, softening, sharpening, or embossing is done it would look like this (see Figure 7A)
Research Question/Problem/ Need	There is a need to be able to quickly and accurately translate from handwritten Gregg Shorthand to digital text with minimal human oversight.
Important Figures	 Sum Transfer Output Path Processing Element Inputs Weights Figure 1 A Basic Artificial Neuron.  Figure 7 a. Weight Matrix W for Character 'a' b.Weight Matrix W for Character 'aback'
Notes	The p-value of 0.002 in the Lilliefors's test for Normality rejects the null hypothesis for the effectiveness of the online and offline processes. The p-value of 0.711 in the Jarque-Bera test which demonstrates neither online nor offline wields an advantage.

Cited references to follow up on	http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.259.4151&rep=rep1&type=pdf
Follow up Questions	- Can this process be generalized to other shorthands? - What characteristics of a shorthand make it easy for the computer to discern?

Article #5 Notes: Segmentation and recognition of phonetic features in handwritten Pitman shorthand

Article notes should be on separate sheets

Source Title	Segmentation and recognition of phonetic features in handwritten Pitman shorthand
Source citation (APA Format)	Ma, Y., Leedham, G., Higgins, C., & Htwe, S. M. (2008). Segmentation and recognition of phonetic features in handwritten Pitman shorthand. *Pattern Recognition*, *41*(4), 1280–1294. https://doi.org/10.1016/j.patcog.2007.10.014
Original URL	https://www.sciencedirect.com/science/article/pii/S0031320307004426
Source type	Journal Article
Keywords	Pitman shorthand, Vocalized outline, Shortform, Segmentation, Classification
Summary of key points (include methodology)	The article started by addressing the fact that in this day and age shorthands are important and relevant due to their usage in digital data entry displays. Pitman's wpm rate makes it the ideal tool to use for the job. First, the article works on analyzing the patterns within the strokes of Pitman Shorthand. It marks the change of direction and also the order to

	check how many times the direction of writing changes. From there, the samples must be segmented so that the flow of different characters combined with each other can be realized and to prep for the Box Model. This Box Model is a helpful tool for AI to break up sensing objects or features in visual data. From here the computer represents the strokes as a series of polar functions and applies the summative vectors and weighted matrices approach standard in Machine Learning
Research Question/Problem/ Need	How do certain features of Pitman shorthand enable it to be understood more or faster to write?
Important Figures	  A B C D ψ_A ψ_B ψ_C ψ_D

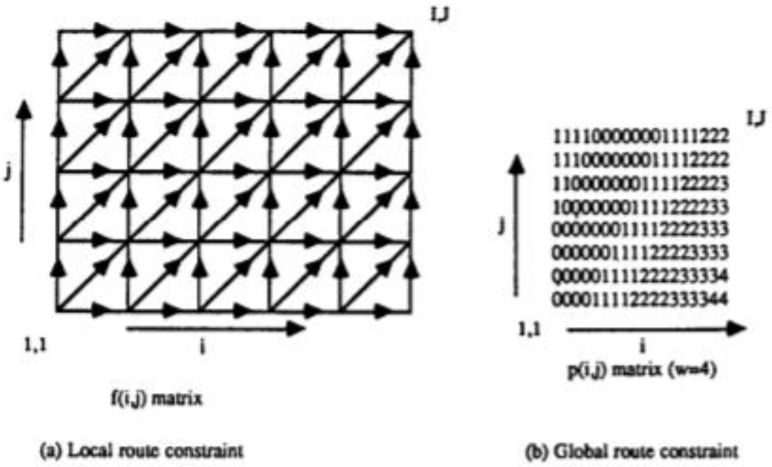
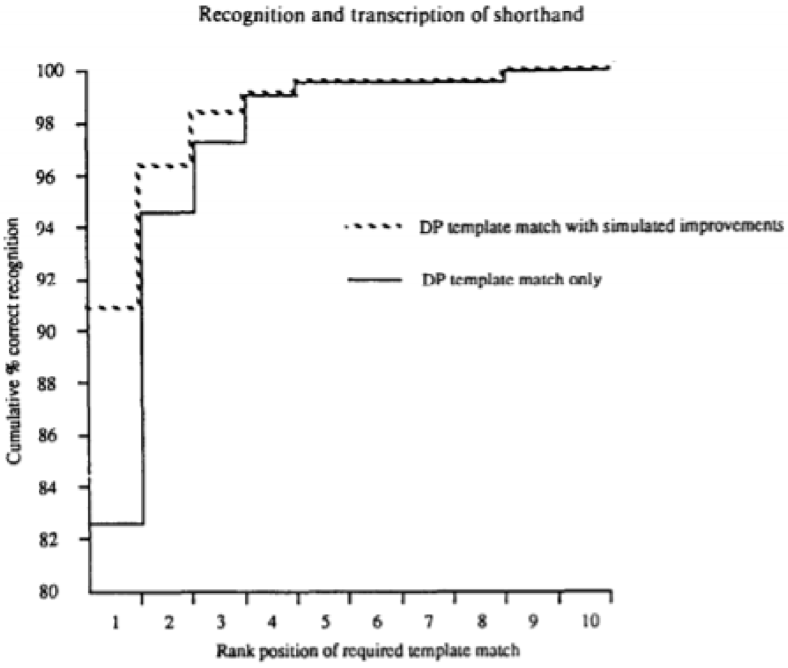
Notes	Experiments on a set of 1127 consonant outlines, 2039 vowels and diphthongs and 841 shortforms have shown that the approaches achieved 75.33%, 96.86% and 91.86% correct recognition accuracy. An experiment on 461 outlines each containing one smooth junction shows that, with the new proposed rule, recognition rate was improved by 55.88% (from 37.53% to 93.41%) at the cost of 14.42% increase in writing time.
Cited references to follow up on	Article Google Scholar Google Scholar
Follow up Questions	How strongly are the intermediate angles related to total sharp movements? How experienced were the professionals? Why was the process of seeing that a sample was legible selected?

Article #6 Notes: Automatic recognition and transcription of Pitman's handwritten shorthand—An approach to shortforms

Article notes should be on separate sheets

Source Title	Automatic recognition and transcription of Pitman's handwritten shorthand—An approach to shortforms
Source citation (APA Format)	*Automatic recognition and transcription of Pitman's handwritten shorthand—An approach to shortforms—ScienceDirect*. (n.d.). Retrieved October 15, 2020, from https://www.sciencedirect.com/science/article/abs/pii/0031320387900082
Original URL	https://www.sciencedirect.com/science/article/pii/0031320387900082
Source type	Journal Article
Keywords	Handwriting, Character recognition, Template match, Dynamic programming, Pitman's shorthand
Summary of key points (include methodology)	There are a series of unofficial yet extremely helpful and efficient changes to Pitman shorthand including new abbreviations and the changes of certain vowel clusters. To track these changes and to measure whether or

	not these changes are significant, they trained an algorithm to read the shorthand and also evaluate its efficiency to write and legibility.
Research Question/Problem/ Need	How can we identify Pitman shorthand by analyzing how it is digitally represented pixelated?

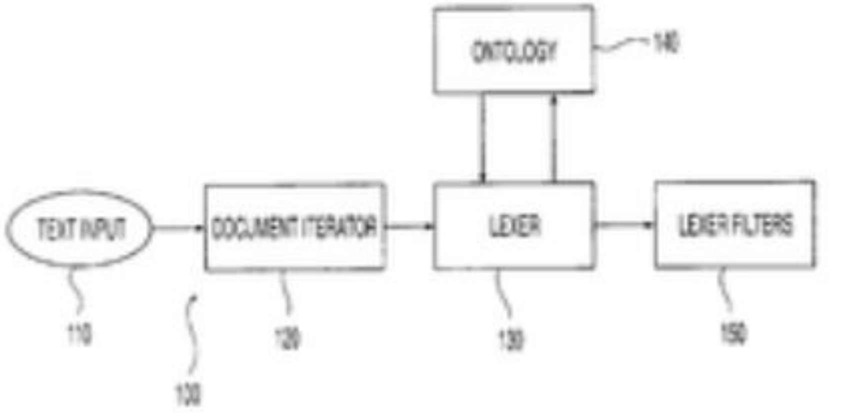
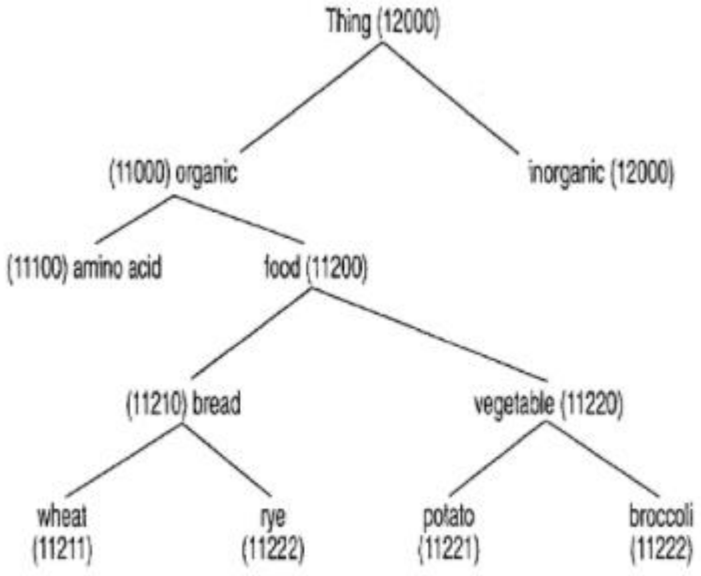
Important Figures	 I,J j 1,1 i f(i,j) matrix I,J 111100000001111222 111000000011112222 110000000111122223 100000001111222233 000000011112222333 000000111122223333 000001111222233334 000011112222333344 j 1,1 i p(i,j) matrix (w=4) (a) Local route constraint (b) Global route constraint  Fig. 6. Cumulative recognition performance for template match.
Notes	They took a similar approach to many matrix based vector analysis procedures as other Handwritten Shorthand Recognition systems. However, the caveat is that in this system it uses a graded scale to gauge it

	more in detail than just zeros and ones. However, this will lead to dealing with heuristics and thus the workload is heavier on the Machine Learning side.
Cited references to follow up on	Segmentation and recognition of phonetic features in handwritten Pitman shorthand Segmentation and recognition of handwritten pitman shorthand outlines using an interactive heuristic search Evaluation of dynamic programming algorithms for the recognition of shortforms in Pitman's shorthand
Follow up Questions	Why didn't this approach use Polar coordinates and instead focused more on the "jaggedness" of the strokes? Is there a reason these particular amendments to the shorthand were selected for this project?

Article #7 Notes: Ontology-based parser for natural language processing

Article notes should be on separate sheets

Source Title	Ontology-based parser for natural language processing
Source citation (APA Format)	*US7027974B1—Ontology-based parser for natural language processing—Google Patents*. (n.d.). Retrieved October 15, 2020, from https://patents.google.com/patent/US7027974B1/en
Original URL	https://patents.google.com/patent/US7027974B1/en
Source type	Patent
Keywords	Lexers, Parsers, Natural Language Processing, predicate-argument
Summary of key points (include methodology)	The natural language processing system aims to loom for specific keywords or types of keywords that may give hints to the literary subject and predicates of sentences. By identifying these attributes, the program can turn these into a format that is more easily readable by search engine and more robotic operators. Ultimately a lexer and parser will need to be used to deal with this new fundamental type of data.
Research Question/Problem/ Need	Search engines, word processors, and other technological tools sometimes have a difficult time understanding human language and thus

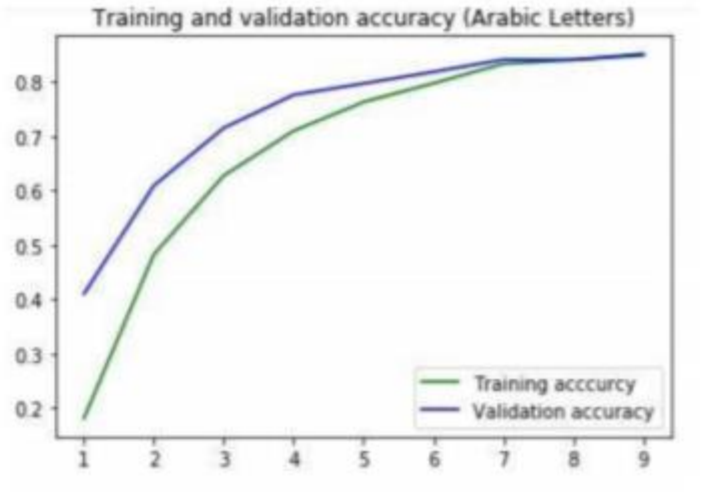
	could stand to benefit from having an algorithm streamline a user’s search phrases.
Important Figures	 ONTOLOGY 140 TEXT INPUT DOCUMENT ITERATOR LEXER LEXER FILTERS 110 100 120 130 150  Thing (12000) (11000) organic inorganic (12000) (11100) amino acid food (11200) (11210) bread vegetable (11220) wheat (11211) rye (11222) potato (11221) broccoli (11222) *Fig. 7*

Notes	The system uses a tagging system in conjunction with a database to sort the words into different classes in a way. By using this method, it starts to create percentages of how likely certain outcomes are to happen. This same sort of tagging technique can perhaps be applied in my project.
Cited references to follow up on	https://patents.google.com/patent/US4270182A/en https://patents.google.com/patent/US4864502A/en https://patents.google.com/patent/US4887212A/en
Follow up Questions	How can this process be further customized for specific search engines? Can this system be directly implemented into search engines?

Article #8 Notes: English-Arabic Handwritten Character Recognition using Convolutional Neural Networks

Article notes should be on separate sheets

Source Title	English-Arabic Handwritten Character Recognition using Convolutional Neural Networks
Source citation (APA Format)	Mohamed, A., & Rohm-Ensing, E. (n.d.). *English-Arabic Handwritten Character Recognition using Convolutional Neural Networks*. 7.
Original URL	http://jmgphd.com/wp-content/uploads/2019/06/handwritten_ocr_report.pdf
Source type	Journal Article
Keywords	HCR (Handwritten Character Recognition), CNN (Convolutional Neural Networks), and image vectors
Summary of key points (include methodology)	The HCR System aimed to create an algorithm that could accurately read the English characters and numerals that the researchers derived from the digital font data set. They did this by using a multiple layer approach with image vectors.
Research Question/Problem/ Need	This is a need for an algorithm to parse through written english arabic (as to not be overfitted to a particular language) and determine its meaning.

Important Figures	 **Figure 6: Chart of accuracy incease over epochs**
Notes	For my project, this can serve as a very important tool for general language recognition because it is generalized to many different types of writing. For example even with my shorthand I developed, there are aspects of english and shorthand that are both used and thus a more generalized solution may help.
Cited references to follow up on	N. Das, A. F. Mollah, S. Saha, and S. S. Haque. Handwritten arabic numeral recognition using a multi layer perceptron. arXiv preprint arXiv:1003.1891, 2010.

	W. Rawat and Z. Wang. Deep convolutional neural networks for image classification: A comprehensive review. Neural computation, 29(9):2352–2449, 2017.
Follow up Questions	How does this account for human error? In what ways can this same system be repurposed for other languages in general? There are definitely databases on kaggle for different handwriting samples, so why didn’t the researchers consider using those for training?

Article #9 Notes: How File Compression Works

Article notes should be on separate sheets

Source Title	How File Compression Works
Source citation (APA Format)	*How File Compression Works | HowStuffWorks*. (n.d.). Retrieved October 15, 2020, from https://computer.howstuffworks.com/file-compression.htm
Original URL	https://computer.howstuffworks.com/file-compression1.htm
Source type	News Article
Keywords	File Compression, Zip Files, LZ Compression
Summary of key points (include methodology)	This article focuses on how applications like WinZip or Stuffit work by targeting the repetitive nature of language. This article then went through how such a program would cut down the size of a famous Kennedy quote: *"Ask not what your country can do for you -- ask what you can do for your country."* First, it tackles this problem by assigning each distinct word a certain numerical value. Then after the reader grasps this basic concept, it moves onto simply identifying any string of characters (including spaces) in order to maximize efficiency. Finally, the article ends with mentioning how color can be

	accommodated to match other already established pixels without compromising resolution.
Research Question/Problem/ Need	How to save file space with a Zip File? How do they work?
Important Figures	The quote has 17 words, made up of 61 letters, 16 spaces, one dash and one period. If each letter, space or punctuation mark takes up one unit of memory, we get a total file size of 79 units. To get the file size down, we need to look for redundancies. The sentence now takes up 18 units of memory, and our dictionary takes up 41 units. So we've compressed the total file size from 79 units to 59 units!
Notes	For my project, I plan to use this same sort of idea once the actual bulk of translating my text file is done and dealt with. While I do not think the last part is applicable, I do however definitely feel that the rest of it can definitely be applied. I am still considering whether or not I have to make a separate dictionary for each translation or should I just make one big one. I will also have to test whether or not it even matters enough in the end thanks to this new LZ adaptive dictionary-based algorithm.

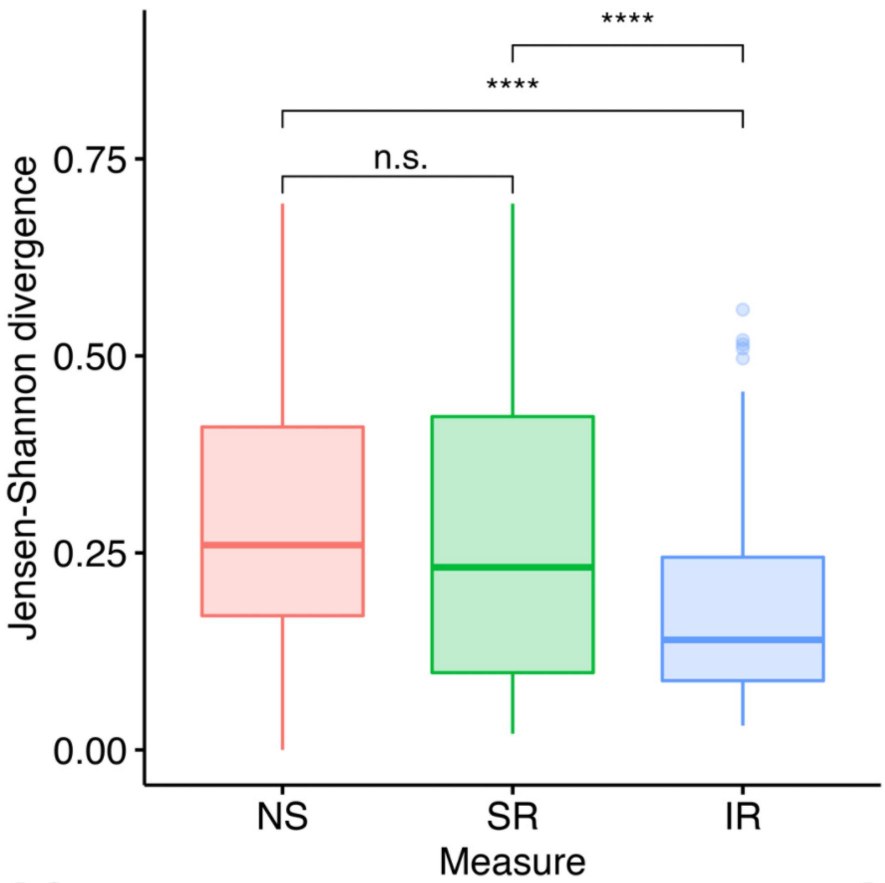
Cited references to follow up on	How does a zip drive store so much more data than a floppy drive? Data-Compression.com
Follow up Questions	Is there a way we can take advantage of the repetitiveness of certain specific individuals? How commonplace is LZ Compression?

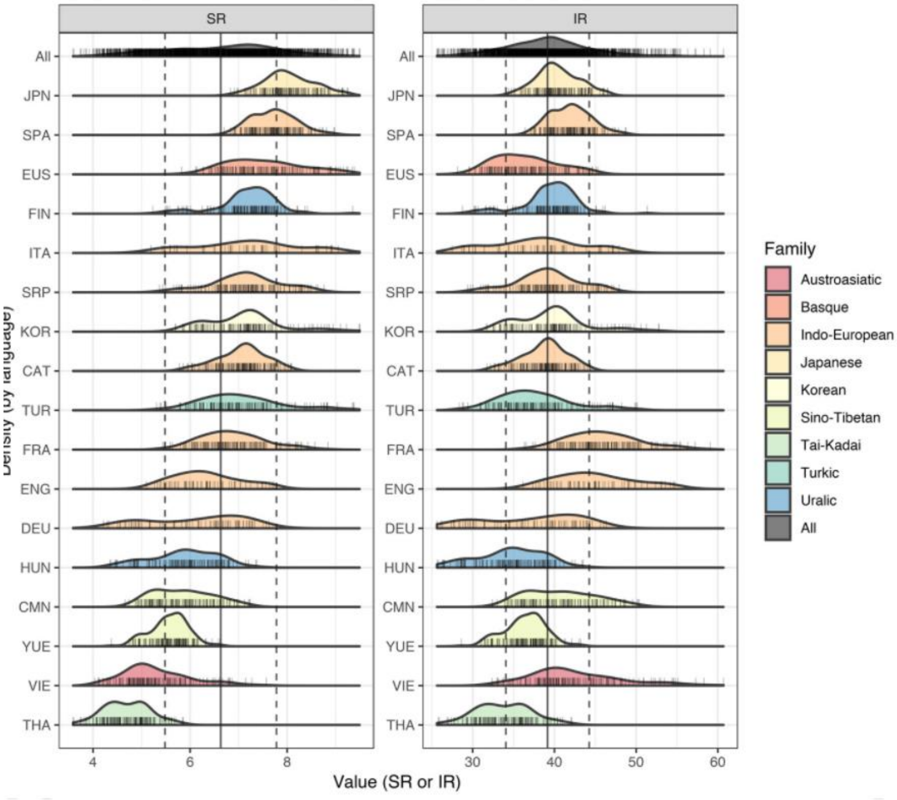
Article #10 Notes: Different Languages, Similar Encoding Efficiency

Article notes should be on separate sheets

Source Title	Different Languages, Similar Encoding Efficiency
Source citation (APA Format)	*Different languages, similar encoding efficiency: Comparable information rates across the human communicative niche | Science Advances*. (n.d.). Retrieved October 15, 2020, from https://advances.sciencemag.org/content/5/9/eaaw2594
Original URL	https://advances.sciencemag.org/content/5/9/eaaw2594
Source type	Encoding, Efficiency, Information Density
Keywords	Languages, Information Density, Linguistic speed
Summary of key points (include methodology)	This article focuses on drawing conclusions between 17 languages across 9 language families to demonstrate the relationships between speech, information density, number of syllables or etc. The project had seemingly taken into consideration several variables that could have contributed to different rates in information and speech such as sex and language family. After a number of mathematical tests and syntagmatic analysis, Figure 1 showed that languages varied greatly in speech while they were remarkably similar in Information rate. Information rate was defined in the experiment as the product of the speech rate and

	information density. Figure 2 showed speech versus information on a 2 dimensional plane with vertical lines. Figure 3 showed that the box and whisker plots of all linguistic aspects and showed that information rate has smaller interquartile ranges.
Research Question/Problem/ Need	Despite the differences in writing and speaking speed of languages, do all languages communicate information at roughly the same rate?

Important Figures	

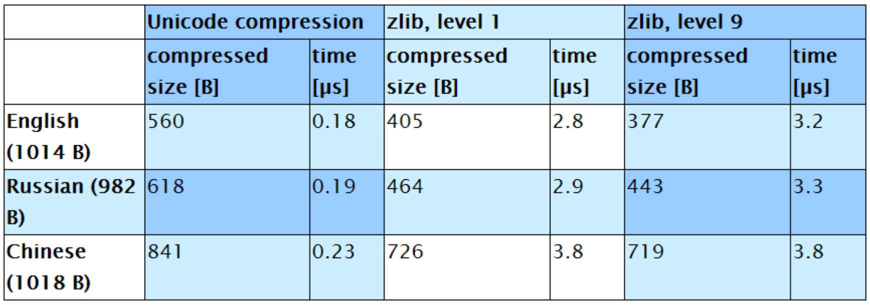
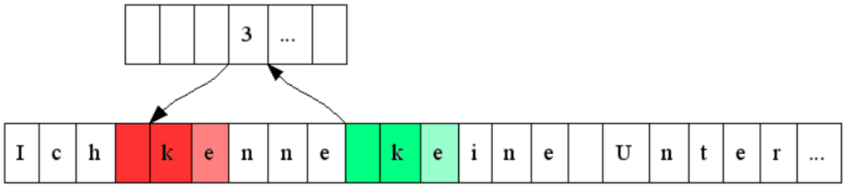
	
Notes	For my project, it is useful to note that Vietnamese did particularly well in this study and thus I will have to consider the possibility of adding the equivalent of a tone in written language. I can also gather from this that although languages in general have a consistent information rate, they have different densities. This gives hope that the information density can be further pushed for this project. I hope to take advantage of this for my project.
Cited references to follow up on	

	S. C. Levinson, Turn-taking in human communication – Origins and implications for language processing. *Trends Cogn. Sci.* 20, 6–14 (2016).CrossRefPubMedGoogle Scholar M. Dingemanse, S. G. Roberts, J. Baranova, J. Blythe, P. Drew, S. Floyd, R. S. Gisladottir, K. H. Kendrick, S. C. Levinson, E. Manrique, G. Rossi, N. J. Enfield, Universal principles in the repair of communication problems. PLOS ONE 10, e0136100 (2015).CrossRefPubMedGoogle Scholar
Follow up Questions	Are there more factors to consider in this experiment including vocabulary size of the language? How are different phonemes and especially linguistic stress taken into account?

Article #11 Notes: Fast Compression Algorithm for UNICODE

Article notes should be on separate sheets

Source Title	Fast Compression Algorithm for UNICODE
Source citation (APA Format)	*Fast Compression Algorithm for UNICODE Text*. (n.d.). Retrieved October 15, 2020, from http://unicode.org/notes/tn31/
Original URL	http://unicode.org/notes/tn31/
Source type	Technical Note
Keywords	Unicode, text compression,
Summary of key points (include methodology)	This article is a more in depth look at the compression process within Unicode itself. This is similar to the first article I read on zip files but this uses slightly different methods. The first method it touches upon is Lempel-Ziv compression that takes previous segments, which may be beneficial depending on how repetitive the sample is. Then they also use a hash table which is a method almost identical to zip files. However it also goes into detail about the storage space of the characters. Compression and decompression is nothing but a matter of assigning values to different segments of the text. There are also some same algorithms and times (in microseconds) for how long this compression and etc should take.

<table>
<tr><td>Research Question/Problem/ Need</td><td>How does Unicode use text compression to make its encoding more efficient?</td></tr>
<tr><td>Important Figures</td><td>

<table>
<tr><th></th><th colspan="2">Unicode compression</th><th colspan="2">zlib, level 1</th><th colspan="2">zlib, level 9</th></tr>
<tr><th></th><th>compressed size [B]</th><th>time [µs]</th><th>compressed size [B]</th><th>time [µs]</th><th>compressed size [B]</th><th>time [µs]</th></tr>
<tr><td>English (1014 B)</td><td>560</td><td>0.18</td><td>405</td><td>2.8</td><td>377</td><td>3.2</td></tr>
<tr><td>Russian (982 B)</td><td>618</td><td>0.19</td><td>464</td><td>2.9</td><td>443</td><td>3.3</td></tr>
<tr><td>Chinese (1018 B)</td><td>841</td><td>0.23</td><td>726</td><td>3.8</td><td>719</td><td>3.8</td></tr>
</table>


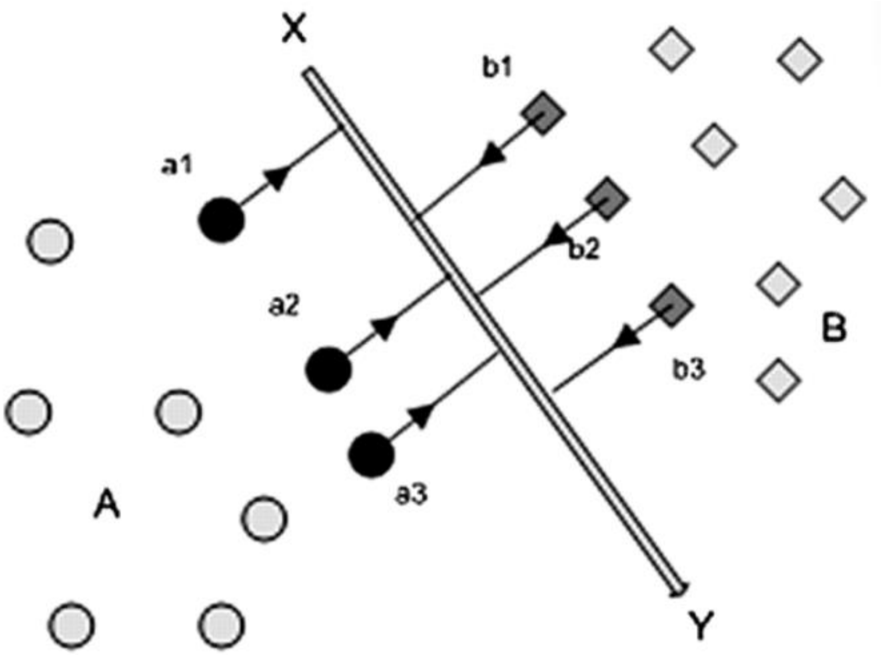
<i>Fig. 2: We get to the position of the previous occurrence by hashing ' ' and 'k' and looking up the pointer in the dictionary.</i>
</td></tr>
<tr><td>Notes</td><td>For the purpose of my project, I will probably end up using hash tables and maybe Lempel-Ziv compression. It will also be helpful to measure the sizes and times given in the table to the times of my own. Hopefully after putting it through my DSK, I can simply use the idea of the code in my program. Unfortunately the sample code is in C and is optimized for understanding instead of efficiency.</td></tr>
<tr><td>Cited references to follow up on</td><td>“What is Unicode” pages from www.unicode.org</td></tr>
</table>

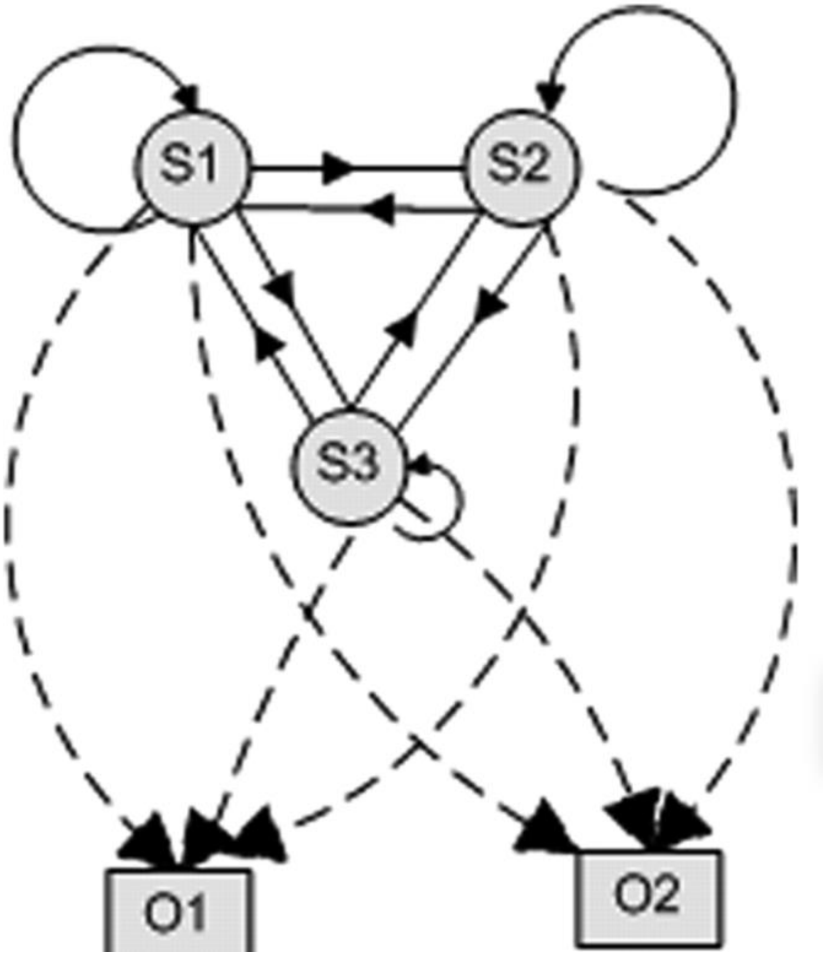
Follow up Questions	Can this system be built upon with other compression methods? Is there a way to make the character encoding quicker for more frequent characters?

Article #12 Notes: Natural Language Processing: An introduction

Article notes should be on separate sheets

Source Title	Natural Language Processing: An introduction
Source citation (APA Format)	*Natural language processing: An introduction | Journal of the American Medical Informatics Association | Oxford Academic*. (n.d.). Retrieved October 15, 2020, from https://academic.oup.com/jamia/article/18/5/544/829676
Original URL	https://academic.oup.com/jamia/article/18/5/544/829676
Source type	Journal Article
Keywords	Natural language processing, Introduction, clinical NLP, knowledge bases, machine learning, predictive modeling, statistical learning, privacy technology
Summary of key points (include methodology)	This article details the beginnings of NLP and how it essentially started from looking at heuristics and complex lexers and parsers to its new analytical form. Then it lists a few different types of NLP designs that can be generally applied to any NLP problem. The article then wraps up by telling the read about IBM Watson and where NLP is heading.

Research Question/Problem/ Need	What are the different ways to tackle NLP using different techniques?
Important Figures	Support Vector Machines  Hidden Markov Models

	
Notes	My algorithm, or at least the referencing in my system, will probably have more backwards propagation and thus will represent the second diagram more. The first diagram deals more with the sorting between two main data types, which while can be applied to the model I propose to build, it is better to get a system that will work more like a function performed on a word.

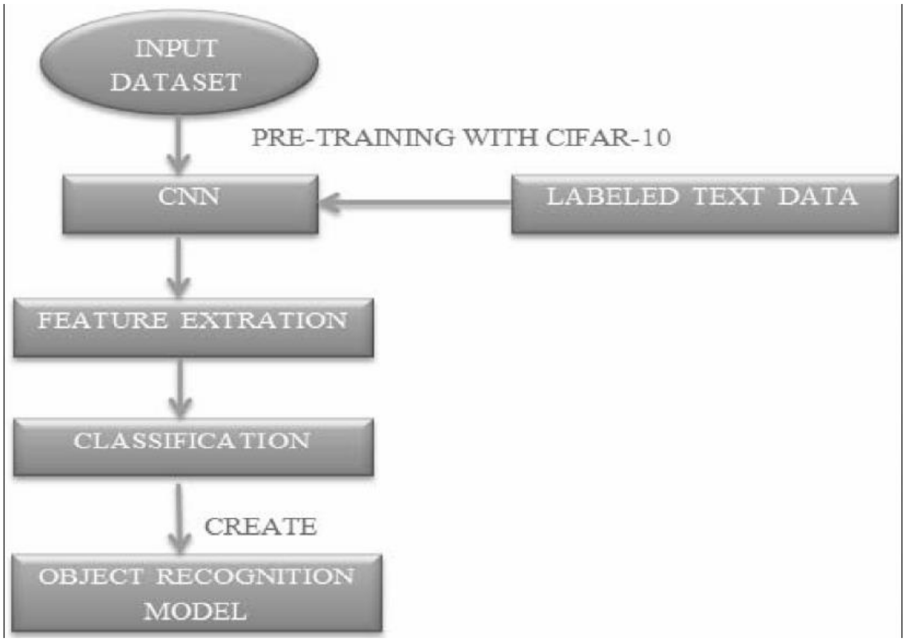
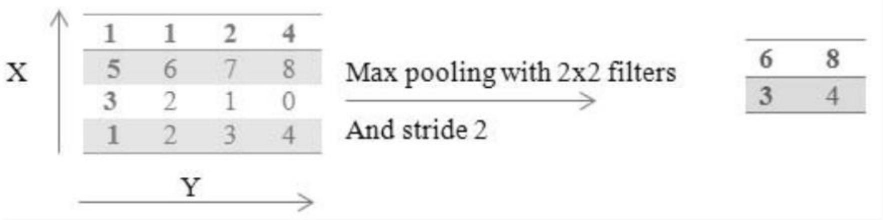
Cited references to follow up on	Manning C Raghavan P Schuetze H . *Introduction to Information Retrieval*. Cambridge, UK: Cambridge University Press, 2008. Hutchins W . *The First Public Demonstration of Machine Translation: the Georgetown-IBM System, 7th January 1954*. 2005. http://www.hutchinsweb.me.uk/GU-IBM-2005.pdf (accessed 4 Jun 2011). Chomsky N . Three models for the description of language. *IRE Trans Inf Theory* 1956;2:113–24.
Follow up Questions	In Figure 4, does the quality of the tasks improve with more intermediate checking, or does it put too much strain on the computer? Would a Markov model or N-grams model be more suitable for my project?

Article #13 Notes: Object recognition in images using convolutional neural network

Article notes should be on separate sheets

Source Title	Object recognition in images using convolutional neural network
Source citation (APA Format)	Sudarshan, D. (n.d.). *Object recognition in images using convolutional neural network—IEEE Conference Publication*. Retrieved December 13, 2020, from https://ieeexplore.ieee.org/document/8398912
Original URL	https://ieeexplore.ieee.org/document/8398912
Source type	Journal Article
Keywords	• Image recognition • Object recognition • Conferences • Control systems • DVD • Convolutional neural networks
Summary of key points (include methodology)	Object detection from repository of images is challenging task in the area of computer vision and image processing in this work we present object classification and detection using cifar-10 data set with intended classification and detection of airplain images. So we used convolutional

	neural network on keras with tensorflow support the experimental results shows the time required to train, test and create the model in limited computing system. We train the system with 60,000 images with 25 epochs each epoch is taking 722to760 seconds in training step on tensorflow cpu system. At the end of 25 epochs the training accuracy is 96 percentage and the system can recognition input images based on train model and the output is respective label of images.
Research Question/Problem/ Need	To what degree of accuracy can convolutional neural networks recognize objects in images from the cifar-10 data set using a Keras library?

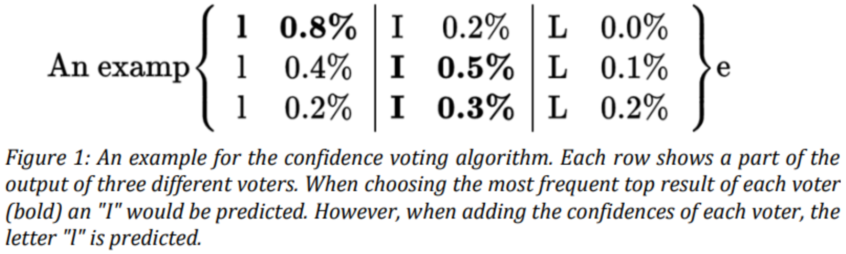
Important Figures	 INPUT DATASET PRE-TRAINING WITH CIFAR-10 CNN LABELED TEXT DATA FEATURE EXTRATION CLASSIFICATION CREATE OBJECT RECOGNITION MODEL  X 1 1 2 4 5 6 7 8 3 2 1 0 1 2 3 4 Y Max pooling with 2x2 filters And stride 2 6 8 3 4 Fig1-2x2 max pooling
Notes	They used a dataset of about a couple thousand images to train their model using multiple layers of the Keras Library They used an existing dataset but a lot of their results were impacted by how well they resized and changed the resolution of the images and etc

Cited references to follow up on	Image Processing and Location based Image Querier(LBIQ) https://ieeexplore.ieee.org/document/9214155
Follow up Questions	What are the pros and cons of Max pooling? How big of a role does Image Preprocessing make here?

Article #14 Notes: Calamari - A High-Performance Tensorflow-based Deep Learning Package for Optical Character Recognition

Article notes should be on separate sheets

Source Title	Calamari - A High-Performance Tensorflow-based Deep Learning Package for Optical Character Recognition
Source citation (APA Format)	Wick, C., Raul, C., & Puppe, F. (n.d.). *[1807.02004] Calamari—A High-Performance Tensorflow-based Deep Learning Package for Optical Character Recognition*. Retrieved December 13, 2020, from https://arxiv.org/abs/1807.02004
Original URL	https://arxiv.org/abs/1807.02004
Source type	Journal Article
Keywords	Tensorflow, Deep Learning, OCR
Summary of key points (include methodology)	Optical Character Recognition (OCR) on contemporary and historical data is still in the focus of many researchers. Especially historical prints require book specific trained OCR models to achieve applicable results (Springmann and Lüdeling, 2016, Reul et al., 2017a). To reduce the human effort for manually annotating ground truth (GT) various techniques such as voting and pretraining have shown to be very efficient

	(Reul et al., 2018a, Reul et al., 2018b). Calamari is a new open source OCR line recognition software that both uses state-of-the art Deep Neural Networks (DNNs) implemented in Tensorflow and giving native support for techniques such as pretraining and voting. The customizable network architectures constructed of Convolutional Neural Networks (CNNS) and Long-ShortTerm-Memory (LSTM) layers are trained by the so-called Connectionist Temporal Classification (CTC) algorithm of Graves et al. (2006). Optional usage of a GPU drastically reduces the computation times for both training and prediction. We use two different datasets to compare the performance of Calamari to OCRopy, OCRopus3, and Tesseract 4.
Research Question/Problem/ Need	How effective is Calamari, a new open source OCR line recognition software that uses state-of-the art Deep Neural Networks (DNNs) implemented in Tensorflow and gives native support for techniques such as pretraining and voting, do for accuracy when applied to read characters from historical documents?
Important Figures	 An examp l 0.8% l 0.4% l 0.2% I 0.2% I 0.5% I 0.3% L 0.0% L 0.1% L 0.2% e *Figure 1: An example for the confidence voting algorithm. Each row shows a part of the output of three different voters. When choosing the most frequent top result of each voter (bold) an "I" would be predicted. However, when adding the confidences of each voter, the letter "l" is predicted.*

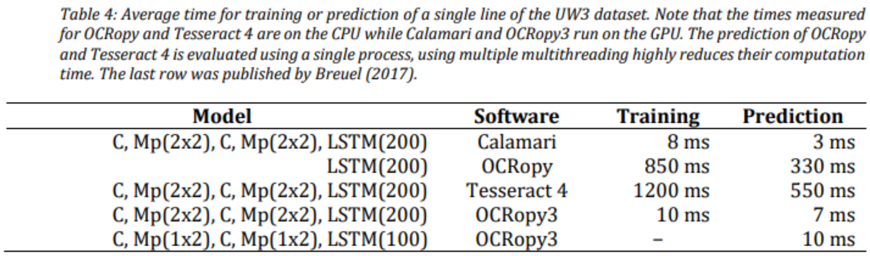


Table 4: Average time for training or prediction of a single line of the UW3 dataset. Note that the times measured for OCRopy and Tesseract 4 are on the CPU while Calamari and OCRopy3 run on the GPU. The prediction of OCRopy and Tesseract 4 is evaluated using a single process, using multiple multithreading highly reduces their computation time. The last row was published by Breuel (2017).

Model	Software	Training	Prediction
C, Mp(2x2), C, Mp(2x2), LSTM(200)	Calamari	8 ms	3 ms
LSTM(200)	OCRopy	850 ms	330 ms
C, Mp(2x2), C, Mp(2x2), LSTM(200)	Tesseract 4	1200 ms	550 ms
C, Mp(2x2), C, Mp(2x2), LSTM(200)	OCRopy3	10 ms	7 ms
C, Mp(1x2), C, Mp(1x2), LSTM(100)	OCRopy3	–	10 ms

Notes

Calamari reaches a Character Error Rate (CER) of 0.11% on the UW3 dataset written in modern English and 0.18% on the DTA19 dataset written in German Fraktur, which considerably outperforms the results of the existing softwares.

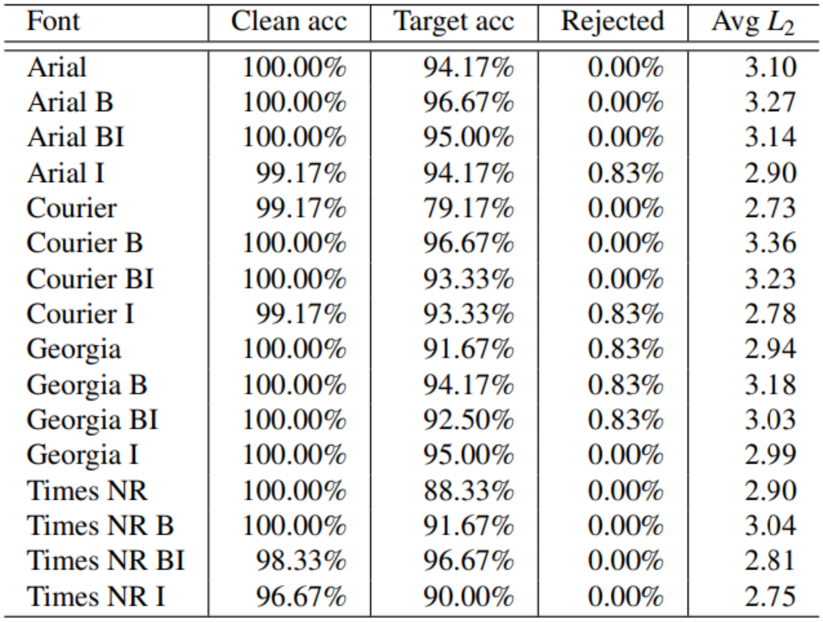
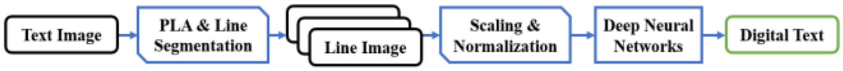
The default network consists of two pairs of convolution and pooling layers with a ReLU-Activation function, a following bidirectional LSTM layer, and an output layer which predicts probabilities for the alphabet. Both convolution layers have a kernel size of 3 × 3 with zero padding of one pixel. The first layer has 64 filters, the second layer 128 filters. The pooling layers implement MaxPooling with a kernel size and stride of 2 × 2. Each LSTM layer (forwards and backwards) has 200 hidden states that are concatenated to serve as input for the final output layer. During training we apply dropout (Srivastava et al., 2014) with a rate of 0.5 to the concatenated LSTM output to prevent overfitting. The loss is computed by the CTC-Algorithm given the output layer’s predictions and the GT label sequence.

Cited references to follow up on	BREUEL, T. M. (2008) The OCRopus open source OCR system. In: Document Recognition and Retrieval XV. International Society for Optics and Photonics, Vol. 6815, p. 68150F. BREUEL, T. M. (2017) High performance text recognition using a hybrid convolutionallstm implementation. In: Document Analysis and Recognition (ICDAR), 2017 14th IAPR International Conference on. IEEE, Vol. 1, pp. 11-16. BREUEL, T. M., et al. (2013) High-performance OCR for printed English and Fraktur using LSTM networks. In: Document Analysis and Recognition (ICDAR), 2013 12th International Conference on. IEEE, pp. 683-687.
Follow up Questions	Why did they have such a different way of presenting their confusion matrix? Their confidence voting seemed to work well but how was it taken into consideration with the confusion matrix?

Article #15 Notes: Fooling OCR Systems with Adversarial Text Images

Article notes should be on separate sheets

Source Title	Fooling OCR Systems with Adversarial Text Images
Source citation (APA Format)	Song, C., & Shmatikov, V. (n.d.). *[1802.05385] Fooling OCR Systems with Adversarial Text Images*. Retrieved December 13, 2020, from https://arxiv.org/abs/1802.05385
Original URL	https://arxiv.org/abs/1802.05385
Source type	Journal Article
Keywords	OCR, adversarial text images, semantic filtering
Summary of key points (include methodology)	We demonstrate that state-of-the-art optical character recognition (OCR) based on deep learning is vulnerable to adversarial images. Minor modifications to images of printed text, which do not change the meaning of the text to a human reader, cause the OCR system to "recognize" a different text where certain words chosen by the adversary are replaced by their semantic opposites. This completely changes the meaning of the output produced by the OCR system and by the NLP applications that use OCR for preprocessing their inputs.

<table>
<tr><td>Research Question/Problem/ Need</td><td>To what extent can OCR be fooled into lowering accuracies and how can we learn from these weaknesses?</td></tr>
<tr><td>Important Figures</td><td>

<table>
<tr><th>Font</th><th>Clean acc</th><th>Target acc</th><th>Rejected</th><th>Avg L_2</th></tr>
<tr><td>Arial</td><td>100.00%</td><td>94.17%</td><td>0.00%</td><td>3.10</td></tr>
<tr><td>Arial B</td><td>100.00%</td><td>96.67%</td><td>0.00%</td><td>3.27</td></tr>
<tr><td>Arial BI</td><td>100.00%</td><td>95.00%</td><td>0.00%</td><td>3.14</td></tr>
<tr><td>Arial I</td><td>99.17%</td><td>94.17%</td><td>0.83%</td><td>2.90</td></tr>
<tr><td>Courier</td><td>99.17%</td><td>79.17%</td><td>0.00%</td><td>2.73</td></tr>
<tr><td>Courier B</td><td>100.00%</td><td>96.67%</td><td>0.00%</td><td>3.36</td></tr>
<tr><td>Courier BI</td><td>100.00%</td><td>93.33%</td><td>0.00%</td><td>3.23</td></tr>
<tr><td>Courier I</td><td>99.17%</td><td>93.33%</td><td>0.83%</td><td>2.78</td></tr>
<tr><td>Georgia</td><td>100.00%</td><td>91.67%</td><td>0.83%</td><td>2.94</td></tr>
<tr><td>Georgia B</td><td>100.00%</td><td>94.17%</td><td>0.83%</td><td>3.18</td></tr>
<tr><td>Georgia BI</td><td>100.00%</td><td>92.50%</td><td>0.83%</td><td>3.03</td></tr>
<tr><td>Georgia I</td><td>100.00%</td><td>95.00%</td><td>0.00%</td><td>2.99</td></tr>
<tr><td>Times NR</td><td>100.00%</td><td>88.33%</td><td>0.00%</td><td>2.90</td></tr>
<tr><td>Times NR B</td><td>100.00%</td><td>91.67%</td><td>0.00%</td><td>3.04</td></tr>
<tr><td>Times NR BI</td><td>98.33%</td><td>96.67%</td><td>0.00%</td><td>2.81</td></tr>
<tr><td>Times NR I</td><td>96.67%</td><td>90.00%</td><td>0.00%</td><td>2.75</td></tr>
</table>


</td></tr>
<tr><td>Notes</td><td>OCR systems are often used as just one component in a bigger pipeline, which passes their output to applications operating on the natural-language text (e.g., document categorization or summarization). These pipelines are a perfect target for the adversarial-image attacks because the output of OCR is not intended to be read or checked by a human. Therefore, the adversary does not need to worry about the syntactic or</td></tr>
</table>

	semantic correctness of the OCR output as long as this output has the desired effect on the NLP application that operates on it.
Cited references to follow up on	[1] ABADI, M., BARHAM, P., CHEN, J., CHEN, Z., DAVIS, A., DEAN, J., DEVIN, M., GHEMAWAT, S., IRVING, G., ISARD, M., ET AL. TensorFlow: A system for large-scale machine learning. In OSDI (2016). [2] Abbyy automatic document classification. https: //www.abbyy.com/en-eu/ocr-sdk/key-features/ classification, 2016.
Follow up Questions	How can this be applied to fit texts from modern times but of subpar quality? Can image preprocessing change this?

Article #16 Notes: Deep Learning Approach in Gregg Shorthand Word to English-Word Conversion

Article notes should be on separate sheets

Source Title	Deep Learning Approach in Gregg Shorthand Word to English-Word Conversion
Source citation (APA Format)	Padilla, D., & Vitug, N. (n.d.). *Deep Learning Approach in Gregg Shorthand Word to English-Word Conversion—IEEE Conference Publication*. Retrieved December 13, 2020, from https://ieeexplore.ieee.org/abstract/document/9177452?casa_token=b7gXAFfpJvwAAAAA:W0wx7xd-H9fJxOqKMsW-CkzFn89081m-ZXRqSJqlAWzfZiBM8oFhLsCdbT2ukpwGFZdl8ao
Original URL	https://ieeexplore.ieee.org/abstract/document/9177452?casa_token=b7gXAFfpJvwAAAAA:W0wx7xd-H9fJxOqKMsW-CkzFn89081m-ZXRqSJqlAWzfZiBM8oFhLsCdbT2ukpwGFZdl8ao
Source type	Journal Article
Keywords	TensorFlow; CNN; Inceptionv3; Gregg Shorthand
Summary of key points	Shorthand or Stenography has been used in a variety of fields of practice, particularly by court stenographers. To record every detail of the hearing, a

(include methodology)	stenographer must write fast and accurate In the Philippines, the stenographers still used the conventional way of writing shorthand, which is by hand. Transcribing shorthand writing is time-consuming and sometimes confusing because of a lot of characters or words to be transcribed. Another problem is that only a stenographer can understand and translate shorthand writing. What if there is no stenographer available to decipher a document? A deep learning approach was used to implement and developed an automated Gregg shorthand word to English-word conversion. The Convolutional Neural Network (CNN) model used was the Inception-v3 in TensorFlow platform, an open-source algorithm used for object classification.
Research Question/Problem/ Need	How well can the Convolutional Neural Network (CNN) model used with the Inception-v3 in TensorFlow platform translate written Shorthand into English?

Important

Figures

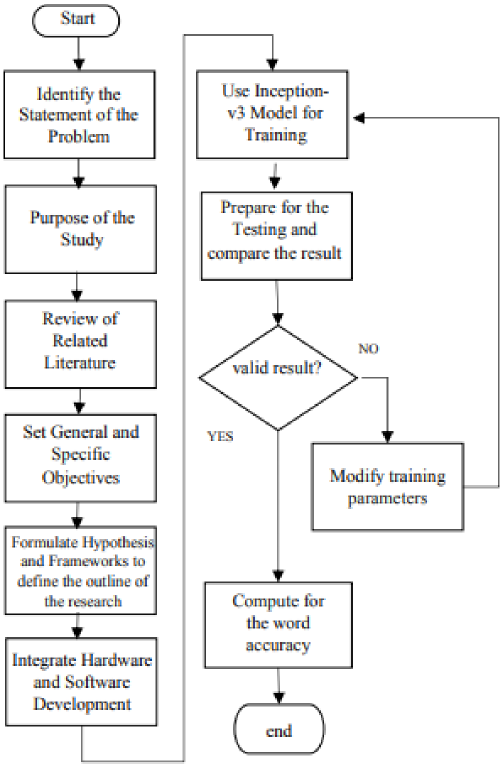
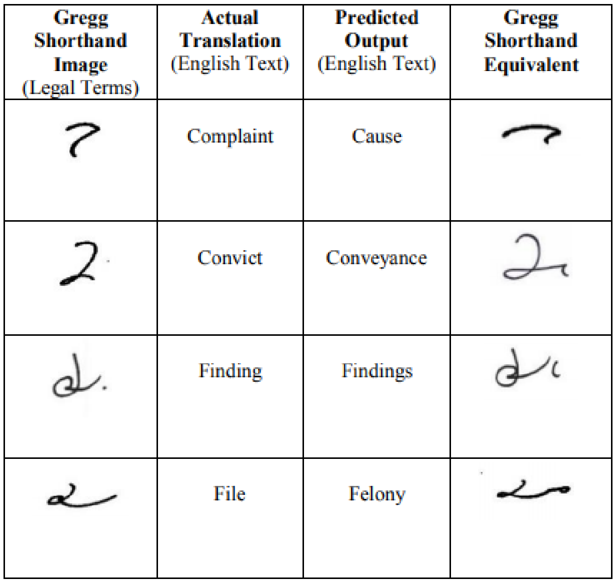

Start
Identify the Statement of the Problem
Purpose of the Study
Review of Related Literature
Set General and Specific Objectives
Formulate Hypothesis and Frameworks to define the outline of the research
Integrate Hardware and Software Development
Use Inception-v3 Model for Training
Prepare for the Testing and compare the result
valid result?
NO
YES
Modify training parameters
Compute for the word accuracy
end

Figure 1. Methodology Framework



Gregg Shorthand Image (Legal Terms)	**Actual Translation** (English Text)	**Predicted Output** (English Text)	**Gregg Shorthand Equivalent**
	Complaint	Cause	
	Convict	Conveyance	
	Finding	Findings	
	File	Felony	

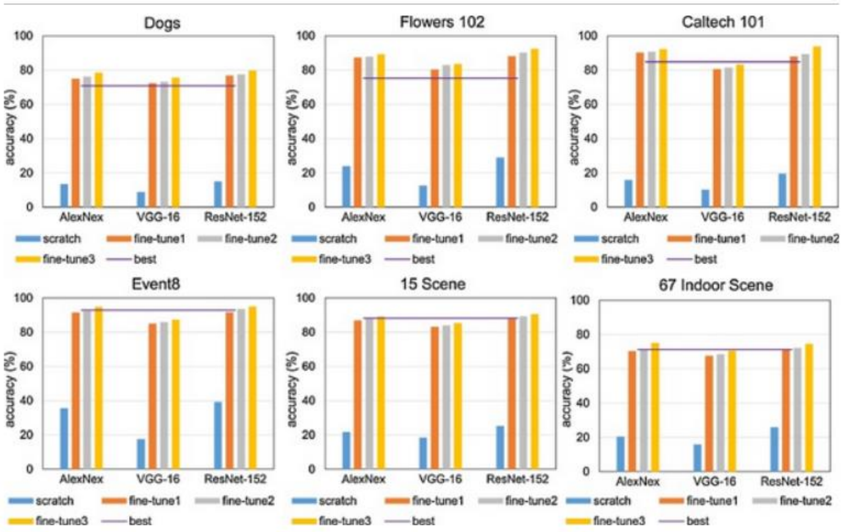
Notes	The training datasets consist of 135 Legal Terminologies with 120 images per word with a total of 16,200 datasets. The trained model achieved a validation accuracy of 91%. For testing, 10 trials per legal terminology were executed with a total of 1,350 handwritten Gregg Shorthand words tested. The system correctly translated a total of 739 words resulting in 54.74% accuracy.
Cited references to follow up on	[1] M. Yang, G. Leedham, C. Higgins, and S. Htwe, "An On-line Recognition System for Handwritten Pitman Shorthand," TENCON 2005 - 2005 IEEE Region 10 Conference, 2005. [2] "Efficiency and Common Problems in Writing Stenography of the Bachelor of South Philippine Adventist College"," South Philippine Adventist College. [Online]. Available: http://www.spaconline.org/home/2016/09/19/efficiency-andcommon-problems-in-writing-stenography-of-the-bachelor-ofscience-in-office-administration-students-of-south-philippineadventist-college/. [Accessed: 09-Sep-2019
Follow up Questions	How does overfitting to a specific field and or region limit this project? How can it be compared to simpler CNN that do not have Deep Learning?

Article #17 Notes: A new image classification method using CNN transfer learning and web data augmentation

Article notes should be on separate sheets

Source Title	A new image classification method using CNN transfer learning and web data augmentation
Source citation (APA Format)	Han, G., Liu, Q., & Fan, W. (n.d.). *A new image classification method using CNN transfer learning and web data augmentation—ScienceDirect*. Retrieved December 14, 2020, from https://www.sciencedirect.com/science/article/abs/pii/S0957417417307844?casa_token=z97ISwPPuZQAAAAA:LEsbigcJl5OM_4KgV3tnQfMyTD15PhK9ul4-oSZfNvJmkBvSN7cR_MoE9wFSuFQAS2ivezI
Original URL	https://www.sciencedirect.com/science/article/pii/S0957417417307844?casa_token=z97ISwPPuZQAAAAA:LEsbigcJl5OM_4KgV3tnQfMyTD15PhK9ul4-oSZfNvJmkBvSN7cR_MoE9wFSuFQAS2ivezI
Source type	Journal Article
Keywords	Feature transferring Data augmentation Convolutional neural network

	Feature representation Parameter fine-tuning Bayesian optimization
Summary of key points (include methodology)	Since Convolutional Neural Network (CNN) won the image classification competition 202 (ILSVRC12), a lot of attention has been paid to deep layer CNN study. The success of CNN is attributed to its superior multi-scale high-level image representations as opposed to hand-engineering low-level features. However, estimating millions of parameters of a deep CNN requires a large number of annotated samples, which currently prevents many superior deep CNNs (such as AlexNet, VGG, ResNet) being applied to problems with limited training data. To address this problem, a novel two-phase method combining CNN transfer learning and web data augmentation is proposed. With our method, the useful feature presentation of pre-trained network can be efficiently transferred to target task, and the original dataset can be augmented with the most valuable Internet images for classification. Our method not only greatly reduces the requirement of a large training data, but also effectively expand the training dataset. Both of method features contribute to the considerable over-fitting reduction of deep CNNs on small dataset. In addition, we successfully apply Bayesian optimization to solve the tuff problem, hyper-parameter tuning, in network fine-tuning. Our solution is applied to six public small datasets. Extensive experiments show that, comparing to traditional methods, our solution can assist the popular deep CNNs to achieve better performance. Particularly, ResNet can outperform all the state-of-the-art

	models on six small datasets. The experiment results prove that the proposed solution will be the great tool for dealing with practice problems which are related to use deep CNNs on small dataset.
Research Question /Problem / Need	Estimating millions of parameters of a deep CNN requires a large number of annotated samples, which currently prevents many superior deep CNNs (such as AlexNet, VGG, ResNet) being applied to problems with limited training data
Important Figures	 Download : Download high-res image (610KB) Download : Download full size image
Notes	• We do image classification on training data limited dataset with deep learning. • Transfer learning is employed to overcome the serious over-fitting. • Web data augmentation is developed to improve the classification performance.

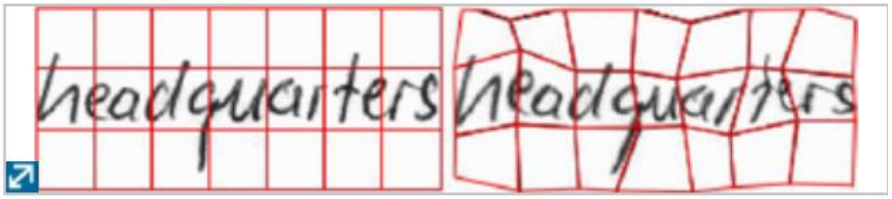
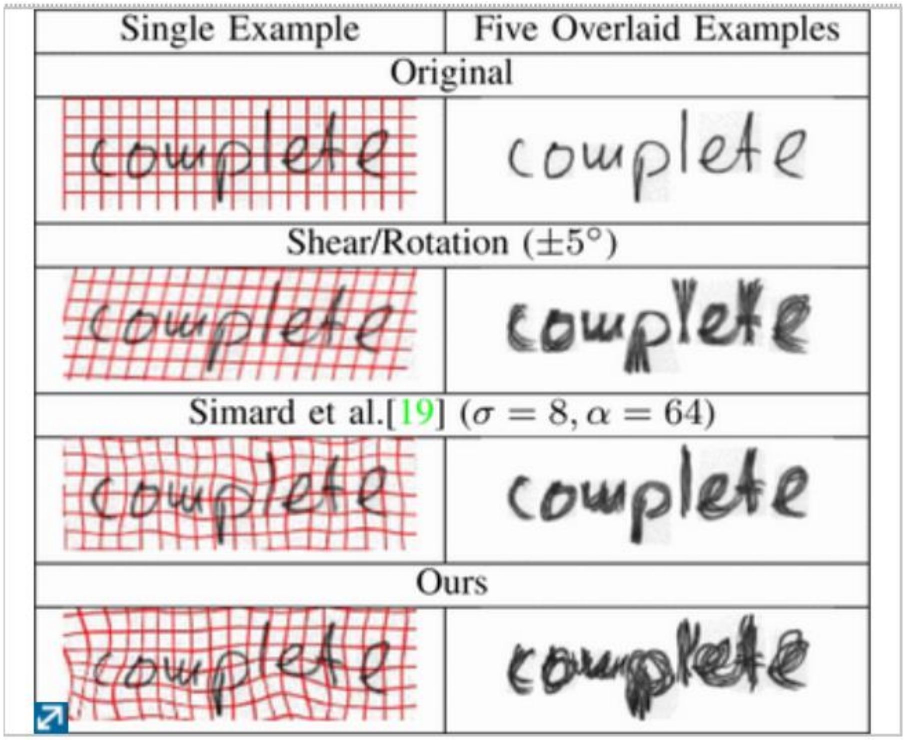
	• Bayesian optimization is employed to facilitate the hyper-parameter search.
Cited references to follow up on	Ando and Zhang, 2005 R.K. Ando, T. Zhang**A framework for learning predictive structures from multiple tasks and unlabeled data** Journal of Machine Learning Research, 6 (November) (2005), pp. 1817-1853 Y.L. Boureau, J. Ponce, Y. LeCun**A theoretical analysis of feature pooling in visual recognition** Proceedings of the 27th international conference on machine learning (ICML-10) (2010), pp. 111-118 .
Follow up Questions	How can the dataset be customized with adapted Bayesian techniques? What causes the differences between the fine-tune groups?

Article #18 Notes: Data Augmentation for Recognition of Handwritten Words and Lines Using a CNN-LSTM Network

Article notes should be on separate sheets

Source Title	Data Augmentation for Recognition of Handwritten Words and Lines Using a CNN-LSTM Network
Source citation (APA Format)	Wigington, C., & Stewart, S. (n.d.). *Data Augmentation for Recognition of Handwritten Words and Lines Using a CNN-LSTM Network—IEEE Conference Publication*. Retrieved December 14, 2020, from https://ieeexplore.ieee.org/abstract/document/8270041?casa_token=v9uOpRqk3-gAAAAA:hv9tt4iFRGKInfLnpYI-0hFJ7kkTBvEV8qUjDN8ynzyc0kis05a5GAE7F__plhOBeBbcPC8
Original URL	https://ieeexplore.ieee.org/abstract/document/8270041?casa_token=v9uOpRqk3-gAAAAA:hv9tt4iFRGKInfLnpYI-0hFJ7kkTBvEV8qUjDN8ynzyc0kis05a5GAE7F__plhOBeBbcPC8
Source type	Journal Article
Keywords	• Data Augmentation , • Handwriting Recognition , • Deep Learning ,

	• Elastic Distortion , • CNN , • LSTM
Summary of key points (include methodology)	We introduce two data augmentation and normalization techniques, which, used with a CNN-LSTM, significantly reduce Word Error Rate (WER) and Character Error Rate (CER) beyond best-reported results on handwriting recognition tasks. (1) We apply a novel profile normalization technique to both word and line images. (2) We augment existing text images using random perturbations on a regular grid. We apply our normalization and augmentation to both training and test images. Our approach achieves low WER and CER over hundreds of authors, multiple languages and a variety of collections written centuries apart. Image augmentation in this manner achieves state-of-the-art recognition accuracy on several popular handwritten word benchmarks.
Research Question/Problem/ Need	How can Data Augmentation, when applied to using a CNN to recognize Gregg shorthand, significantly improve performance?

Important Figures	 **Fig. 3.** Word image with uniform grid superimposed. 2nd image (right) with distorted grid and image distorted accordingly.  Single Example Five Overlaid Examples Original Shear/Rotation (±5°) Simard et al.[19] (σ = 8, α = 64) Ours
Notes	Study mostly focused on English and German Study was very extensive on image preprocessing The data augmentation tactics utilized a wide repertoire including everything from rotation to “scrunchiness”

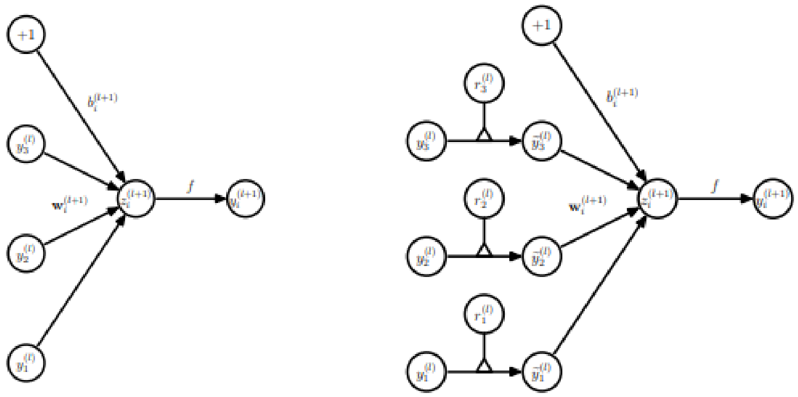
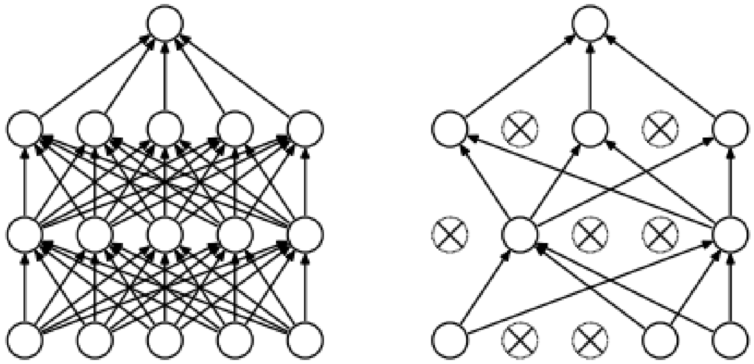
Cited references to follow up on	A. Poznanski and L. Wolf, "Cnn-n-gram for handwriting word recognition", *Proc. CVPR 2016*. P. Krishnan, K. Dutta and C. V. Jawahar, "Deep feature embedding for accurate recognition and retrieval of handwritten text", *The 15th International Conference on Frontiers in Handwriting Recognition (ICFHR)*, 2016.
Follow up Questions	Is it time efficient to use this type of image preprocessing? Can deep learning look beyond data augmentation?

Article #19 Notes: Dropout: a simple way to prevent neural networks from overfitting

Article notes should be on separate sheets

Source Title	Dropout: a simple way to prevent neural networks from overfitting
Source citation (APA Format)	Srivasta, N., Hinton, G., & Krizhevsky, A. (n.d.). *Dropout: A simple way to prevent neural networks from overfitting: The Journal of Machine Learning Research: Vol 15, No 1*. Retrieved December 14, 2020, from https://dl.acm.org/doi/abs/10.5555/2627435.2670313
Original URL	https://dl.acm.org/doi/abs/10.5555/2627435.2670313
Source type	Journal Article
Keywords	Computing Methodologies, Machine Learning, Neural Networks, Dropout
Summary of key points (include methodology)	Deep neural nets with a large number of parameters are very powerful machine learning systems. However, overfitting is a serious problem in such networks. Large networks are also slow to use, making it difficult to deal with overfitting by combining the predictions of many different large neural nets at test time. Dropout is a technique for addressing this problem. The key idea is to randomly drop units (along with their connections) from the neural network during training. This prevents units

	from co-adapting too much. During training, dropout samples from an exponential number of different "thinned" networks. At test time, it is easy to approximate the effect of averaging the predictions of all these thinned networks by simply using a single unthinned network that has smaller weights. This significantly reduces overfitting and gives major improvements over other regularization methods. We show that dropout improves the performance of neural networks on supervised learning tasks in vision, speech recognition, document classification and computational biology, obtaining state-of-the-art results on many benchmark data sets.
Research Question/Problem/ Need	How effective is Random Dropout at developing a ML Model?
Important Figures	

	 +1 $b_i^{(l+1)}$ $y_3^{(l)}$ $\mathbf{w}_i^{(l+1)}$ $z_i^{(l+1)}$ f $y_i^{(l+1)}$ $y_2^{(l)}$ $y_1^{(l)}$ $r_3^{(l)}$ $r_2^{(l)}$ $r_1^{(l)}$ $\tilde{y}_3^{(l)}$ $\tilde{y}_2^{(l)}$ $\tilde{y}_1^{(l)}$ (a) Standard network (b) Dropout network Figure 3: Comparison of the basic operations of a standard and dropout network.  (a) Standard Neural Net (b) After applying dropout.
Notes	Dropout neural networks can be trained using stochastic gradient descent in a manner similar to standard neural nets. The only difference is that for each training case in a mini-batch, we sample a thinned network by dropping out units. Forward and backpropagation for that training case are done only on this thinned network. The gradients for each parameter are averaged over the training cases in each mini-batch. Any training case which does not use a parameter contributes a gradient of zero for

	that parameter. Many methods have been used to improve stochastic gradient descent such as momentum, annealed learning rates and L2 weight decay. Those were found to be useful for dropout neural networks as well.
Cited references to follow up on	M. Chen, Z. Xu, K. Weinberger, and F. Sha. Marginalized denoising autoencoders for domain adaptation. In *Proceedings of the 29th International Conference on Machine Learning*, pages 767-774. ACM, 2012. G. E. Dahl, M. Ranzato, A. Mohamed, and G. E. Hinton. Phone recognition with the mean-covariance restricted Boltzmann machine. In *Advances in Neural Information Processing Systems 23*, pages 469-477, 2010.
Follow up Questions	How can we facilitate this process to target specific areas of the model?

Article #20 Notes: Combining Convolutional Neural Network With Recursive Neural Network for Blood Cell Image Classification

Article notes should be on separate sheets

Source Title	Combining Convolutional Neural Network With Recursive Neural Network for Blood Cell Image Classification
Source citation (APA Format)	Liang, G., & Hong, H. (n.d.). *Combining Convolutional Neural Network With Recursive Neural Network for Blood Cell Image Classification—IEEE Journals & Magazine*. Retrieved December 13, 2020, from https://ieeexplore.ieee.org/abstract/document/8402091
Original URL	https://ieeexplore.ieee.org/abstract/document/8402091
Source type	Journal Article
Keywords	Artificial intelligence, convolutional neural network, recurrent neural network, transfer learning.
Summary of key points (include methodology)	The diagnosis of blood-related diseases involves the identification and characterization of a patient's blood sample. As such, automated methods for detecting and classifying the types of blood cells have important medical applications in this field. Although deep convolutional neural

	network (CNN) and the traditional machine learning methods have shown good results in the classification of blood cell images, they are unable to fully exploit the long-term dependence relationship between certain key features of images and image labels. To resolve this problem, we have introduced the recurrent neural networks (RNNs). Specifically, we combined the CNN and RNN in order to propose the CNN-RNN framework that can deepen the understanding of image content and learn the structured features of images and to begin endto-end training of big data in medical image analysis. In particular, we apply the transfer learning method to transfer the weight parameters that were pre-trained on the ImageNet dataset to the CNN section and adopted a custom loss function to allow our network to train and converge faster and with more accurate weight parameters. Experimental results show that compared with the other CNN models such as ResNet and Inception V3, our proposed network model is more accurate and efficient in classifying blood cell images.
Research Question/Problem/ Need	How can we automate the process of identifying red-blood cells with AI?

Important Figures

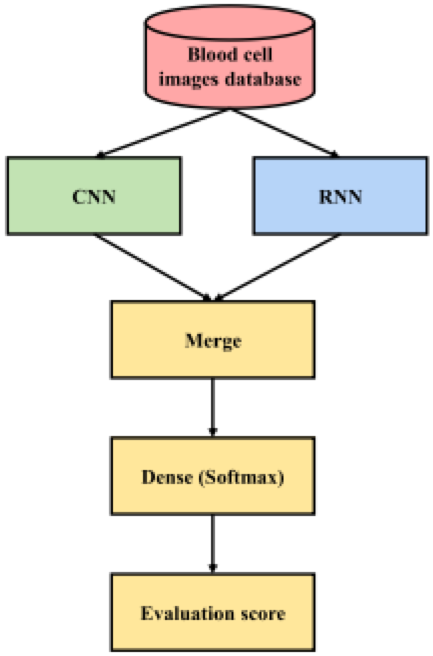


FIGURE 1. Overview of the proposed method using CNN-RNN framework and transfer learning for classifying blood cell images.

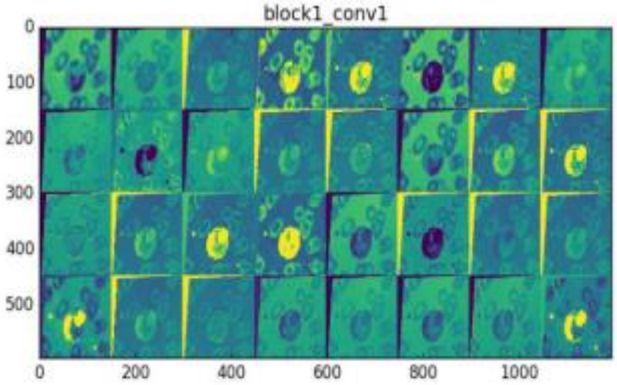
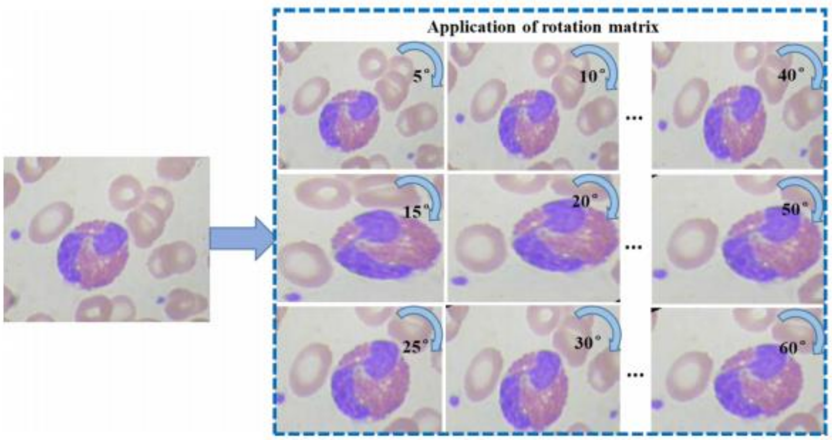


FIGURE 7. Visualization of the result of the activation (feature maps) of in the first convolutional layer of the Xception-LSTM fine-tuned on the dataset (block1_conv1 is the first convolutional layer of our model).

	 Application of rotation matrix 5° 10° ... 40° 15° 20° ... 50° 25° 30° ... 60° **FIGURE 4.** Image patches are generated from the blood cell image by application of rotation matrix so that the acquired images can be cropped to further augment the dataset.
Notes	They had a rotation matrix and focused heavily on one specific type of image preprocessing (or 2) They had a standard confusion matrix This was a color model and not in grayscale like many projects.
Cited references to follow up on	[1] N. Sinha and A. G. Ramakrishnan, ‘‘Automation of differential blood count,’’ in Proc. Conf. Convergent Technol. Asia–Pacific Region (TENCON), Bengaluru, India, vol. 2, 2003, pp. 547–551. [2] P. Yampri, C. Pintavirooj, S. Daochai, and S. Teartulakarn, ‘‘White blood cell classification based on the combination of eigen cell and parametric feature detection,’’ in Proc. 1st IEEE Conf. Ind. Electron. Appl., Singapore, May 2006, pp. 1–4.

Follow up Questions	How would its effectiveness be limited without color? Would its effectiveness be improved?