# George Mason University

# DAEN 690

# Summer 2020

# **Accure Autoencoder for Denoising OCR**

**Report – Full Sprint 5**

**Team Autoencoder**

**Gauthami Karavi**

**Jun Wang**

**Matt Machado**

**Stephen Schade**

**Yun Li**

**8/17/2020**

Table of Contents

[Abstract 3](#_Toc48489565)

[1 Introduction 4](#_Toc48489566)

[1.1 Background 4](#_Toc48489567)

[1.2 Problem Space 4](#_Toc48489568)

[1.3 Research 4](#_Toc48489569)

[1.4 Solution Space 5](#_Toc48489570)

[1.5 Project Objectives 5](#_Toc48489571)

[1.6 Primary User Story 5](#_Toc48489572)

[1.7 Product Vision 6](#_Toc48489573)

[Scenario #1 6](#_Toc48489574)

[Scenario #2 6](#_Toc48489575)

[1.8 Definition of Terms: 6](#_Toc48489576)

[2 Data Acquisition 7](#_Toc48489577)

[2.1 Overview: 7](#_Toc48489578)

[2.2 Field Descriptions: 7](#_Toc48489579)

[2.3 Data Context: 7](#_Toc48489580)

[2.4 Data Conditioning 8](#_Toc48489581)

[2.5 Data Quality Assessment: 8](#_Toc48489582)

[2.6 Other Data Sources 8](#_Toc48489583)

[3 Analytics and Algorithms 9](#_Toc48489584)

[3.1 Convolutional Neural Network 9](#_Toc48489585)

[3.2 Noisy Autoencoder 10](#_Toc48489586)

[3.3 Convolution 11](#_Toc48489587)

[4 Visualization 12](#_Toc48489588)

[5 Findings 12](#_Toc48489589)

[6 Summary 13](#_Toc48489590)

[7 Future Work 13](#_Toc48489591)

[8 Appendix 13](#_Toc48489592)

[8.1 Appendix - Github 13](#_Toc48489593)

[8.2 Appendix – Project Risk Matrix 13](#_Toc48489594)

[8.3 Appendix – Agile Workflow Summary 14](#_Toc48489595)

[9 References 14](#_Toc48489596)

# Abstract

Optical Character Recognition (OCR) is a process in which text from scanned images is converted to computer enabled characters to be used in further computational analysis. This process works best on documents that contain highly visible and clean images to reliably extract information. However, not all documents are ready made to be scanned into OCR and require additional pre-processing steps to enhance the image sufficiently. These ‘noisy’ documents contain such elements as stains, marks, crumples, and other items. The objective of this project is to build an autoencoder capable of taking noisy images and removing these elements to improve OCR recognition. With a dataset from RVL-CDIP at Carnegie Melon University, our autoencoder is trained using a Convolutional Neural Network (CNN). This neural network learns through the process of convolution and pooling how to remove pixel information that is least relevant to reconstructing the final image. These weights from the trained model are used in the autoencoder to predict an enhanced version of the input image which improves the ability for characters to be properly recognized via OCR. **[NEEDS CONCLUSION ADDED]**

# Introduction

## Background

There have always been obstacles to transitioning from a paper-based world into the new digital era with computers and technology. Specifically, there are vast amounts of old books, manuscripts, receipts, papers, etc. that are currently sitting in boxes waiting to be brought into a new life. The issue in translating these old documents into digital form is dealing with potential errors in the documents themselves. Google was the first to pioneer a methodology for translating problem texts due to image noise into computer enabled text using a service called reCAPTCHA. The need for this service came about due to Google’s project to digitalize books via the Google Books Project. They discovered that approximately 20% of texts from scanned physical books could not be properly transcribed by computers and they needed a way to properly capture all the information present. The way reCAPTCHA works is by displaying errored words from OCR to users to verify if they are human or not. These words were displayed directly as an image and if enough humans wrote the same word; then this word would be entered into the digitalized version of the item transcribed. This of course takes effort from human users to scale up to the needs of the sheer volume of physical items that need to be digitalized. As of September 2008, reCAPTCHA had transcribed 440 million problemed words from OCR with 99% accuracy. [1]

## Problem Space

Optical Character Recognition (OCR) technology enables the extraction of text data from images. There are endless uses for this type of technology, from the digital processing of historical documents to the extraction of text in personal photos. It is designed to capture numbers, letters, and punctuation and transfer these characters into data that can be searched, processed, analyzed, and stored.

OCR technology is valuable in any instance where documents or images have text. This is especially useful as society moves away from physical documents and into the digital data space. Using OCR technology, entire document archives once relegated to file cabinets can instead be harvested for digital text. Doing so creates opportunities for non-digital text to be leveraged by other computer-enabled technologies.

This technology is not perfect, and it is difficult to translate character images into a digital text without some inaccuracies or errors. Image “noise,” which can occur if image resolution or coloring issues interfere with the characters, is one reason that OCR technology may have trouble reading character text.

## Research

Research efforts for this project were broken down into two areas, Convolutional Neural Networks (CNN) and Autoencoder. The primary method of research was utilizing internet searches either Google or George Mason University library services to view webpages, research papers, and books written.  
  
For Convolution Neural Networks, the primary resource involved a book called Practical Convolutional Neural Networks [2]. This gave an outline of CNNs and their applications either via supervised classification methods or unsupervised Autoencoders. Using the basis of this book, the team was able to identify an intuitive understand of how these networks derived value to apply to our Autoencoder. Additionally, the resource Gentile Dive into Math Behind Convolutional Neural Networks [3] provided insight into the mathematics of CNNs. This assisted the team in understanding the behind the scenes math that supported the code and the methodology applied in this model.

For Autoencoders, a critical aspect was reviewing other projects that had similar project objectives to our problem statement. The main dataset in which CNN had been applied was called MNIST, which contained 28x28 input images on different characters. This MNIST dataset is the standard on which most current Autoencoders are tested on and is outlined by the resource Towards Data Science: Convolutional Autoencoders for Image Noise Reduction [4]. This resource helped formed a standard in which to compare our image denoising based upon a well-established dataset and methodology. The team utilized this resource as a basis of the derived code for our Autoencoder model.

## Solution Space

Building upon the efforts of reCATPCHA into digitalizing text from the pre-computer age, a better solution is needed to operate at sufficient scale utilizing new automated technologies. Google’s effort requires manual users across the world to use their service repeatedly to generate sufficient data to accurately transcribe a noisy word. Our Autoencoder is intended to be automated solution at scale where a user could dump hundreds of thousands of scanned documents into a database and the Autoencoder can quickly improve the image quality and eliminate noise. This does not require any further intervention by the user and the new OCR results can be obtained quickly and with confidence that the final product delivers the corrected output.

## Project Objectives

This project aims to build a Convolutional Neural Network (CNN) auto-encoder that reduces image noise during the OCR process. We will evaluate the fidelity of our CNN by conducting OCR text extraction on a dataset before and after the implementation of our method. The project will use K-means clustering, Python’s Tesseract library (i.e., PyTesseract), and denoise imager. We plan to leverage Google Colab and AWS SageMaker computing resources. In addition to delivery of a prototype, the project team will evaluate our denoising model with respect to accuracy and performance. [5]; [6]; [7]; [8]; [9]; [10]; [11]

This project assumes the availability of compute resources and the instruction and guidance of commercial and academic stakeholders. Additionally, our ability to assess the usefulness of the constructed auto-encoder relies on the availability of training data considered to be ‘noisy’ regarding standard OCR methods.

## Primary User Story

Our team has developed the following user story to assist in the guidance of our project:

“As a user, we want to be able to process all data into a something that a computer can easily digest for further use cases. However, not all data is currently in this format and therefore requires products to make this process easier. For our users with an interest in taking older scanned documents and extracting data, they may run into several issues involving noise such as coffee stains, markings, or wrinkles that could impede OCR technology from performing optimally. Our user would need a product that could accurately and efficiently remove this noise and increase the ability of the OCR technology to capture the data into a computational form for further analysis. We image our product helping these users to enhance their OCR and data acquisition efforts.”

## Product Vision

### Scenario #1

One example of a use case for this product is to take old corporate SEC filings from the era in which typewriters were primarily the form of creating documents and transform this potential noisy text into useful text for computers. These documents may have markings that prevent a computer from being able to read into an OCR technology and valuable data could be lost in the translation. Unlike current OCR technology, our autoencoder is a pre-processing machine and ensures information is captured appropriately. Without an autoencoder, the process would have to entail a lot of manual reading and typing by a human which is costly in both man hours as well as invoking human error into the process. A trained autoencoder could provide confidence that historical data captured via OCR technology is useful for furthering the development of algorithms that determine the proper value of a security.

### Scenario #2

Another use care of this product is for a business that has medical records of patients before any Electronic Medical Record (EMR) was implemented. These patients’ records could have been typed in with a typewriter or could be old enough to sustain damage such as water stains and wrinkles. Doctors may want to use our product to scan these documents into computer text to assist them in understanding a patient’s medical history. These older medical records can then be integrated with current EMR technologies and used in variety of analysis such as medical research and enhancing diagnostics.

## Definition of Terms:

**OCR** (Optical Character Recognition): A technology that is able to recognize the printed and handwritten character from images in a real document.

**PyTesseract**: A wrapper for leveraging OCR. It can process different types of images, such as jpeg, png, gif, tif, etc.

**CNN** (Convolutional Neural Network): A deep learning algorithm which consists of one or more convolutional layers and one or more fully connected layers, which is similar with a standard multilayer neural network. CNN can detect and classify images.

**Encoder**: A network that accepts inputs and outputs feature vectors and can keep the feature and information of inputs.

**Decoder**: A network that takes the feature vectors from the encoder and outputs a result that is closest to the expected output.

**Kernel Convolution**: The process of applying a digital matrix to a part of an image and transforming it according to the values in the matrix.

**Rectified Linear Activation Function (ReL)**: A activation function introduces the nonlinear factor into the neural network, through which the neural network can fit various curves.

**Sparse Categorical Crossentropy**: A type of loss function which is appropriate when the classes are mutually exclusive.

# Data Acquisition

## Overview:

This project uses the Ryerson Vision Lab Complex Document Information Processing (RVL-CDIP) dataset [12]. This dataset is the product of academic research by Adam W. Harley, Alex Ufkes, and Konstantinos G. Derpanis. Their 2015 paper, “Evaluation of Deep Convolutional Nets for Document Image Classification and Retrieval,” researched the ability of convolutional neural networks to conduct state-of-the-art document image classification and retrieval [13]. RVL-CDIP is a subset of the IIT-CDIP Test Collection, which was used for a 2006 conference paper titled “Building a test collection for complex document information processing [14].” The IIT-CDIP dataset is itself a subset of the Legacy Tobacco Document Library (TLDL), an archive of 14 million documents related to tobacco industry advertising, manufacturing, marketing, scientific research, and political activities. The full dataset is hosted by the University of California, San Francisco university library [15].

## Field Descriptions:

The data consist of 400,000 grayscale images and 16 classes with 25,000 images per class. All images are unique. Data is classified using the following labels.

|  |  |
| --- | --- |
| 0 – letter | 8 – file folder |
| 1 – form | 9 – news article |
| 2 – email | 10 - budget |
| 3 - handwritten | 11 - invoice |
| 4 - advertisement | 12 - presentation |
| 5 – scientific report | 13 – questionnaire |
| 6 – scientific publication | 14 - resume |
| 7 - specification | 15 - memo |

## Data Context:

This dataset was used by researchers at Carnegie Mellon University in 2015 to build a convolutional neural network focused on document image, classification, and retrieval.

## Data Conditioning

The data is stored as a downloadable file on Google Drive using a link provided on the dataset website. The data is zipped into a .tar file with two separate data sources, one for the images and one that provides information on labeling.

## Data Quality Assessment:

The RVL-CDIP dataset is an academic creation that is designed to lend itself to computer vision and machine learning efforts. For each image, the largest dimension is not to exceed 1000 pixels. All are image formats and are classified by type of document as well as a predefined train, test, and validation set.

The raw dataset has several outstanding issues that need to be mitigated before final analysis can occur. The first issue is that each image in the dataset is separated into individual file folders making it difficult to extract the image into a tidy corpus for analysis. The second issue is that the data is currently in .TIF format (Tagged Image File) and may be insufficient for analysis in further steps. The third issue presented is the sheer size of the dataset corresponding with the preferred data storage on Google Drive. Google Drive uploads the information slowly and either the data needs to be scaled down for analysis only or another tool needs to be research for proper uploading. Finally, the dataset contains a few “bad” images for our analysis. These images contain little or no text and almost all noise and need to be eliminated from our purposes.

* Completeness: The RVL-CDIP dataset includes 400,000 grayscale images in 16 classes, with 25,000 images per class. Also, it is split into 320,000 training images, 40,000 validation images, and 40,000 test images.
* Uniqueness: The dataset consists 16 different classes, and each image in each class is unique.
* Consistency – Images are sized so not exceed 1000 pixels for largest dimension.
* Integrity – All images are .tif (Tagged Image File), which may be challengeable for analysis in further steps.
* Conformity – All images are stored in a standard format.
* Accuracy – Data appears to be accurately managed given the quality of the source.
* Overall Quality: The quality of the dataset is high.

## Other Data Sources

We have identified several additional datasets that may also be used for our project. These datasets vary in quality, volume of data, and usefulness toward our defined project scope.

# Analytics and Algorithms

## Convolutional Neural Network

This autoencoder model will be built using a Convolutional Neural Network (CNN) with unsupervised learning to reduce noise in various text images. A regular neural network contains three layers: an input layer, a hidden layer, and an output layer. The input layer is simply a vector of data that is passed through a series of hidden layers to produce a result in the output layer. A traditional neural network is show below:

A close up of a logo

Description automatically generated

Figure 1: A Neural Network

Source: Adapted from [2]

A Convolution Neural Network has a similar structure but differs by having three distinct layers: the convolutional layer, the pooling layer, and the fully connected layer. CNN’s also almost always include images so spatial information and their relation to each other at the pixel level is important. This is the key aspect of the convolutional layer which reduces the image to smaller pixel vectors while retaining the relationship of those pixels to each other. This process is called filtering and the more filters used in the model the longer the model takes to produce a result. The next layer is called the pooling layer which takes information from the convolutional layer and reduces the pixels even further. The process by which it reduces the image is by using max pooling by looking for which pixels have the highest value between them. The higher the value is, the more important the pixel is to the image. The convolution and pooling layers continue until finally being flattened into one vector called the fully connected layer [2]. An example of a basic CNN is below:

A picture containing clock, drawing

Description automatically generated

Figure 2: A Convolutional Neural Network (Layers)

Source: Adapted from [2]

## Noisy Autoencoder

The autoencoder will use CNN’s to train and reduce image noise by reconstructing the image into an improved version. This means there are two steps in the training process. The first step is shrinking the images down into flattened vectors with the most important features remaining. This step is the encoder step. The second step, or decoding step, is to take the flattened vectors from the encoding step as an input. These inputs are then reshaped and ran through the convolutional and pooling layers to recreate an improved image. The pooling layers in the decoder instead of reducing pixels by max pooling increase the size of the image using Up sampling. This process produces an output of the improved original image. An example of the Autoencoder process is below:

A close up of text on a white background

Description automatically generated

Figure 3: Autoencoder Overview

Source: Adapted from [4]

## Convolution

Pixels are the building blocks of digital imagery. In digital graphics, the pixel is the basic and smallest unit. In simplest terms, pixels are colored squares. Each square’s color is usually based on the intensity of basic or primary colors.

Pixels have no defined size. Instead, the quantity of pixels in a digital image is defined by its resolution, or how many pixels are found in a certain area. For example, a five-megapixel image (or consisting of five million pixels) may occupy the same area as a 10-megapixel image, but the 10-megapixel image would have twice the resolution as the five-megapixel image. Convolution attempts to take entire sections of colored pixels and translating it into numbers that can be used by a computer. [16]



Figure 4: Digital Image Data Structure [3]

Kernel Convolution is the process of applying a matrix of numbers to portions of the image, transforming it as you go based on the values in the matrix. This creates a feature map of the image by using the below formula (see Figure 5). The imput image (f) and the matrix, known as a kernel (h), are applied to the pixel vector space rows and columns (m and n). As the function steps through the pre-defined row-column space each time, the pixel values are matrix multiplied with the kernel value and the resulting number is added to the now-reduced feature map (see Figure 6). This process is considered one convolution. [3]

Figure 5: Convolution equation

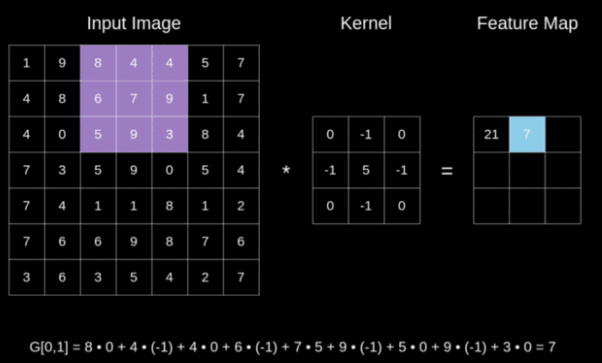


Figure 6: Digital Image Data Structure [3]

Next, in order to make use of the new feature map, we need to apply an activation function. The role of the activation function is to turn the summed feature map into an output for that node. There are several activation functions used in CNNs. We chose to use the rectified Linear activation function (ReL), which is a nonlinear function that uses stochastic gradient descent with backpropagation. ReL is popular because of it’s built-in gradient descent/backpropagation learning methods, but also because it acts both linear and nonlinear in nature. For values greater than 0, the function performs linearly, making it easier to understand and optimize. However, it also acts nonlinear for values less than 0 by creating a resulting output of 0. The simple statement, “If input > 0, Return input; else Return 0” sums up the idea behind ReL. A convolutional node that uses the ReL function is known as a Rectified Linear Unit (ReLU). Neural networks that leverage RELU are often called Rectified Networks. [17]

Each convolution in a neural network follows the same process, but each additional convolution does not have to use the same rules. These tunable features, or hyperparameters, are used to refine and direct the convolutional process based on need. One such parameter, Step length, or Stride, sets the rules for how many pixels the kernel will shift for each calculation. This will effectively set the spatial dimensions of the feature map for that convolution. The filter size, or Kernel size, is also considered a hyperparameter. [3]

# Visualization

**[Describe and show findings and results based on a mix of figures and descriptive text. If you have video, it will be limited to presentation, however, it can also be reference as media file in your Blackboard file exchange.]**

# Findings

**[FINDINGS]**

# Summary

***[Summarize your findings and results for the reader. What did you discover, prove, disprove, etc.]***

# Future Work

The project recommends three main items to build upon the findings of this paper for future work. The first item is to improve the results of the Autoencoder by splitting the images into smaller pieces and running a CNN on those smaller images. This not only will improve the speed at which the images are processed but also has the potential to improve the denoising process. The second item is to continue adjustment of the parameters to optimize and improve the neural network. There are a multitude of parameters to adjust such as filter size, latent dimension space, number of training images, number of layers and optimizers. Experimentation is critical to building a successful Autoencoder and continued efforts to improve and optimize could result in a more sophisticated model to achieve the desired results. Finally, in our dataset there were 16 different document types and any future model could investigate and build models for each document type. These document types have different use cases and could add additional value to the end user. There is also a reasonable chance that certain document types could have worked better using the existing model and further experimentation would help determine a better solution.

**[LESSONS LEARNED?]**

# Appendix

## Appendix - Github

Project documents, code and research can be located at the following Github repository:

<https://github.com/Mmachad/autoencoder>

## Appendix – Project Risk Matrix

A screenshot of a cell phone

Description automatically generated

## Appendix – Agile Workflow Summary

This Project was tracked and managed using an Agile development workflow application called YouTrack. The summary of the project sprints, dates, and task accomplished is below:



# References

|  |  |
| --- | --- |
| [1] | L. v. Ahn, B. Maurer, C. McMillen, D. Abraham and M. Blue, "reCAPTCHA: Human-Based Character Recognition via Web Security Measures," *Science Magazine,* 12 September 2008. |
| [2] | R. a. P. P. Karim, Practical Convolutional Neural Networks, Packt Publishing, 2018. |
| [3] | P. Skalaski, "Gentile Dive into Math Behind Convolutional Neural Networks," Towards Data Science, 12 April 2019. [Online]. Available: https://towardsdatascience.com/gentle-dive-into-math-behind-convolutional-neural-networks-79a07dd44cf9. [Accessed 11 July 2020]. |
| [4] | D. Dataman, "Towards Data Science: Convolutional Autoencoders for Image Noise Reduction," 20 November 2019. [Online]. Available: https://towardsdatascience.com/convolutional-autoencoders-for-image-noise-reduction-32fce9fc1763. [Accessed 6 July 2020]. |
| [5] | C. G. K. S. I. Wiraatmaja, "The Application of Deep Convolutional Denoising Autoencoder for Optical Character Recognition Preprocessing," in *International Conference on Soft Computing, Intelligent System and Information Technology*, 2017. |
| [6] | L. H. X. Zheng, "Character Segmentation for License Plate Recognition by K-Means ALgorithm," *G. Maino and G.L. Foresti (Eds.): ICIAP,* vol. II, no. LNCS 6979, pp. 444-453, 2011. |
| [7] | J. L. Wei, "Autoencoders: Neural Networks for Unsupervised Learning," Intuitive Deep Learning : Medium, 18 Febuary 2019. [Online]. Available: https://medium.com/intuitive-deep-learning/autoencoders-neural-networks-for-unsupervised-learning-83af5f092f0b. [Accessed 6 June 2020]. |
| [8] | A. Rosebrock, "Using Tesseract OCR with Python," Pyimagesearch.com, 10 July 2017. [Online]. Available: https://www.pyimagesearch.com/2017/07/10/using-tesseract-ocr-python/. [Accessed 6 June 2020]. |
| [9] | J. P. S. G. S. G. H. S. D. Shriansh Srivastava, "Optical Character Recognition on Bank Cheques Using 2D Convolution Neural Network," *Applications of Artificial Intelligence Techniques in Engineering. Advances in Intelligent Systems and Computing,* vol. 697, pp. 589-596, 2018. |
| [10] | A. I. A. Muna Ahmed Awel, "REVIEW ON OPTICAL CHARACTER RECOGNITION," *International Research Journal of Engineering and Technology,* vol. 6, no. 6, p. June, 2019. |
| [11] | M. M. N. M. Roland Graef, "A Novel Hybrid Optical Character Recognition Approach for Digitizing Text in Forms," *Lecture Notes in Computer Science,* vol. 11491, 2019. |
| [12] | A. U. a. K. G. D. Adam W. Harley, "The RVL-CDIP Dataset," Carneigie Mellon University Department of Computer Science, 2020. [Online]. Available: https://www.cs.cmu.edu/~aharley/rvl-cdip/. [Accessed 21 June 2020]. |
| [13] | A. U. a. K. G. D. Adam W. Harley, "Evaluation of Deep Convolutional Nets for Document Image Classification and Retrieval," *ICDAR,* 2015. |
| [14] | G. A. S. A. O. F. D. G. a. J. H. D. Lewis, "Building a test collection for complex document information processing," in *Proc. 29th Annual Int. ACM SIGIR Conference*, 2006. |
| [15] | University of California, San Francisco, "The Legacy Tobacco Document Library (LTDL)," 2007. [Online]. Available: http://legacy.library.ucsf.edu/.. [Accessed 21 June 2020]. |
| [16] | A. Mireles, "How to Understand Pixels, Resolution, and Resize Your Images in Photoshop Correctly," Digital Photography School, [Online]. Available: https://digital-photography-school.com/understand-pixels-resolution-resize-photoshop/. |
| [17] | J. Brownlee, "A Gentle Introduction to the Rectified Linear Unit (ReLU)," Machine Learning Mastery, 9 January 2019. [Online]. Available: https://machinelearningmastery.com/rectified-linear-activation-function-for-deep-learning-neural-networks/. [Accessed 11 July 2020]. |