8-2018

Word Recognition in Nutrition Labels with Convolutional Neural Network

Anuj Khasgiwala
Utah State University

WORD RECOGNITION IN NUTRITION LABELS WITH CONVOLUTIONAL NEURAL NETWORK

by

Anuj Khasgiwala

A thesis submitted in partial fulfillment
of the requirements for the degree

of

MASTER OF SCIENCE

in

Computer Science

Approved:

Vladimir Kulyukin, Ph.D. Major Professor	Xiaojun Qi, Ph.D. Committee Member
Haitao Wang, Ph.D. Committee Member	Mark R. McLellan, Ph.D. Vice President for Research and Dean of the School of Graduate Studies

UTAH STATE UNIVERSITY
Logan, Utah

2018

ABSTRACT

Word Recognition in Nutrition Labels with Convolutional Neural Network

by

Anuj Khasgiwala, Master of Science

Utah State University, 2018

Major Professor: Vladimir Kulyukin, Ph.D.
Department: Computer Science

An imperative part of a healthy eating regimen is the understanding and maintenance of nutritional data and comprehension of how extraordinary food items and nutrition constituents influence our bodies.

This dissertation is the descriptive form of developing a procedure to encapsulate Deep Learning along with Computer Vision based algorithm for extracting nutritional information from nutritional labels (NLs) accessible on most packaged food items which is most important for proactive nutrition management. In light of the fact that it enhances the user's ability to capture continuous nutrition information accumulation and analysis.

The system′s front end will be a smartphone application which will be used to capture an image of nutrition table on any packaged food item. The system′s back end is a system where images are first processed in Java using OpenCV, vision based algorithms and wavelet algorithms. The image obtained are then sent to Deep Convolution Neural Network for word classification of all nutrition labels (NL).

(49 pages)

PUBLIC ABSTRACT

Word Recognition in Nutrition Labels with Convolutional Neural Network

Anuj Khasgiwala

Nowadays, everyone is very busy and running around trying to maintain a balance between their work life and family, as the working hours are increasing day by day. In such a hassled life people either ignore or do not give enough attention to a healthy diet. An imperative part of a healthy eating routine is the cognizance and maintenance of nourishing data and comprehension of how extraordinary sustenance and nutritious constituents influence our bodies. Besides in the USA, in many other countries, nutritional information is fundamentally passed on to consumers through nutrition labels (NLs) which can be found in all packaged food products in the form of nutrition table. However, sometimes it turns out to be challenging to utilize this information available in these NLs notwithstanding for consumers who are health conscious as they may not be familiar with nutritional terms and discover it hard to relate nutritional information into their day by day activities because of lack of time, inspiration, or training. So it is essential to automate this information gathering and interpretation procedure by incorporating Machine Learning based algorithm to abstract nutritional information from NLs on the grounds that it enhances the consumer's capacity to participate in nonstop nutritional information gathering and analysis.

ACKNOWLEDGMENTS

I would like to utilize this opportunity to express my most profound thanks to my major professor, Dr. Vladimir Kulyukin, for his advice, guidance, and patience throughout my graduate level education. His recommendations and suggestions have been invaluable to me. His experience and knowledge is the secret behind the accomplishment of the project.

I would also like to acknowledge my gratitude to my graduate committee members, Dr. Xiaojun Qi and Dr. Haitao Wang for all their help and suggestions on this dissertation.

Finally, I appreciate the continuous support and encouragement of my beloved parents and my brother throughout the duration of my academic pursuits.

Anuj Khasgiwala

CONTENTS

LIST OF TABLES

LIST OF FIGURES

ACRONYMS

NL	Nutrition label
NLEA	Nutrition Education and Labelling Act
OCR	Optical Character Recognition
TSAW	Text Skew Angle Wavelets
2DHWT	2D Haar Wavelet Transform
CNN	Convolutional Neural Network
DNN	Deep Neural Network
JBIG	Joint Bi-level Image Experts Group
FDA	Food and Drug Administration
ML	Machine Learning
No.	Number

CHAPTER 1

INTRODUCTION

1.1 Background

According to a study done by U.S. Department of Agriculture, it was estimated that, US resident′s calorie intake has grown on an average by 523 calories per day in last 30 years [1]. World Health Organization indicated in their annual report, obesity causes maladies such as diabetes, kidney failures, and strokes, and predicts these diseases will be one of the dominating factor for death in the whole world [2]. It has been considered that unorganized diet is the reason behind 30-35 percent of cancer cases. The health-care has evolved so much recently but still there is no long lasting cure found yet for diabetes and cancer, but most nutritionists and dietitians suggest nutrition management as the key to forestall diseases to their patients. Therefore having knowledge of nutrition label (NLs) is required to have a proper diet. Besides in the USA, in many other countries, putting nutrition information is required on every packaged food items by the Nutrition Education and Labeling Act (NLEA) of 1990 [3, 4].

There are 18 core nutrition labels, each of which contains a lot of information. People make important decisions regarding their health with the information they infer from these labels. [5]. This information is exhibited in the form of a standard table. Although, once in a while the information in these labels might be hard to decipher by the common user who may find it hard to discover and understand nutritional terms on numerous packaged items [6].

Evolution in the computer science can be the key in understanding procedure and give consumers the ability to pick what is ideal for them by making the nutritional information more easy for client understanding. In a grocery store environment, a easily accessible method of innovation is the mobile phone which remains extremely underutilized. The

gigantic processing power of these gadgets can be put to use to extricate nutritional information from NLs accessible on most packaged items, utilizing computer vision based procedures to implement a proactive nutrition management framework which will enhance the user's capacity to participate in consistent nutritious information accumulation and analysis.

1.2 Nutrition Label Tracking

Vision based identification of NLs will use the cell phone cameras to capture images of NLs for ensuing handling and extraction of nutrition information. This incorporates restricting the NL in the picture and utilizing Deep Learning to peruse the text information from the NL. Skew angle is the angle that the text lines in the digital image makes with the horizontal direction. Text in such cases is rotated or twisted and corrupts the performance of further processing and may genuinely influence the performance of consequent phases of division and acknowledgment, since the contemporary Optical Character Recognizer (OCR) frameworks can not deal with pivoted text and perform well just in perceiving text that are linearly adjusted. While the horizontally adjusted text is effortlessly identified and perceived, skewed text represents a provocation to recognition.

1.2.1 Image Segmentation and OCR

Text Skew Angle Wavelets (TSAW) algorithm was designed to use 2D Haar Wavelet Transform (2DHWT). This algorithm takes nutrition table as input and applies numerous iterations of 2DHWT to calculate horizontal, vertical and diagonal change matrices of image. After this, the image is chunked into multiple images and then passed to OCR to convert images to text. But this algorithm and process flow had lots of drawbacks or limitations. The OCR does not work well with rotated images or images which are not vertical. So we added a feature of rotating the image to make it vertical before passing it to OCR. The other limitation of using OCR, Tesseract, was the font style because the OCR are not trained for font style Helvetica which is the standard font style used for nutrition table, due to which the OCR gets confused to transform image into text like, 'g' as '9'. This issue

was resolved by training Tesseract using a tool, named, jTessBoxEditor. This improved the accuracy of Tesseract, but still the accuracy was not enough. The TSAW algorithm was unable to process and segment images in few scenarios like, images that were already vertical. This algorithm was unable to segment images if there is lots of noise available in the nutrition table images passed as input. We did analysis and made some alterations to TSAW for noise reduction.

1.2.2 Deep Learning

Applying different changes on image segmentation and OCR for converting images to text, increased the accuracy of nutrition label prediction, but after these changes the accuracy became stagnant and no further changes were helped in increasing the accuracy.

Machine Learning has always attracted all technically evolving sectors by analyzing the trend and predicting the outcome. The increase in research and use of machine learning algorithms in different scenarios has opened doors for innovations and high accuracy.

Machine learning algorithms can be categorized into 2 categories: Supervised and Unsupervised learning. Our scenario of classifying the word comes under supervised learning category because we have both, train and validation data labeled and we are trying to find a function to map input with output.

Designing any machine learning algorithm has been a very complicated task because it requires to find the significant and required features. To surpass this limitation, a new system was developed called Deep Learning. Deep learning automatically finds the best features needed for prediction.

Convolution Neural Network has been specially designed for image classification and it has been proven that CNN is the best for images by achieving very high accuracy. Lots of Universities and companies are performing their research using deep learning and CNN for driver-less cars, image recognition, natural language processing etc.

1.3 Process Flow

The process flow used in this research consist of following steps:

1. Capture image of nutrition tables using smartphone.
2. The captured image is then passed to modified TSAW algorithm which segment the image into multiple images. These images consists of different NLs.
3. The segmented images are then rotated to vertical if required.
4. The rotated images are passed to Tesseract, so that, we can have a a ground truth to compare with CNN.
5. The rotated images are manually cropped into region of interest and then labeled.
6. The labeled images are passed to CNN for training and validation.
7. CNN is then able to classify images into different NL.

The complete process flow is converted into pictorial form shown in Fig. 1.1.

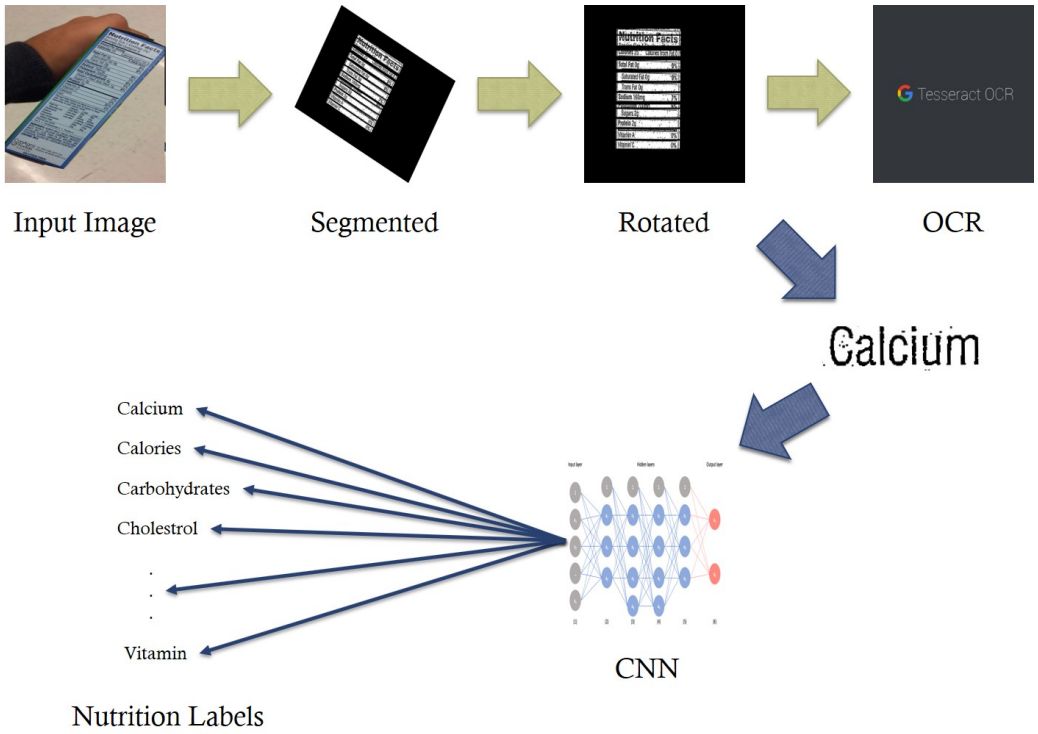

Fig. 1.1: Flow Chart

1.4 Research Scope

The main role of this research is to verify the hypothesis that nutrition data can be extricated utilizing integration of vision based procedures and deep learning from pictures of packaged food items, which were taken with the help of a hand-held smart-phone camera in a supermarket.

This research exhibits an algorithm that introduces a framework that can be utilized to distinguish the skew angle of a NL with no imperatives on the rotation magnitude and in this manner extricate nutrition information from it by utilizing Deep Learning.

This research was done in 2 stages. The first stage was the advancement of TSAW algorithm. The second and the final stage was to implement a Deep CNN for classification of images into nutrition labels.

The remaining dissertation is organized as follows: In chapter 2, we give some foundation information and talk about related work. In chapter 3, we discuss about the advancement brought to TSAW algorithm and OCR for more accuracy. In chapter 4, we have tried to talk about how Deep Learning has changed the world and how it will help us get higher accuracy. In chapter 5, we will try to describe all the experiments performed to design CNN and the results. In chapter 6, we have shared the inferences we have formed, the reason for less accuracy and future scope.

CHAPTER 2
RELATED WORK

2.1 Introduction

In this chapter we will discuss the foundation and related works related to nutrition detection and tracking. We partition the chapter into two areas viz. related work on text skew angle detection, reading, and use of Machine Learning (ML) for nutrition tracking.

2.2 Text Skew Angle Detection - Background

Vision-based object detection has been the center of many researches and innovative projects from a very long time. An assortment of algorithms have been created to determine the text skew angle. They can be classified by the methodologies that they take to solve the problem.

The primary class of algorithms normally utilize horizontal or vertical projection profiles. A horizontal and vertical projection profile is a 1-dimensional array whose size is equivalent to the number of rows and number of columns in the image respectively. Every location in a projection profile stores a count of number of black pixels related with text in the corresponding row or column of the image. Projections can be thought of as a 1D histograms.

The idea of projection profiles was spearheaded [7] and subsequently patented by Postl [8]. Postl's algorithm utilizes the horizontal projection profile for text skew angle detection. The algorithm computes the horizontal projection profiles for angles in the vicinity of 0 and 180 degrees in little additions. It utilizes the aggregate of squared differences between contiguous components of the projection profile as the criterion function and picks the profile that boosts that esteem.

Hull [9] proposes a text skew angle detection algorithm similar to Postl′s. Hulls algorithm is more effective, in light of the fact that it rotates individual pixels instead of rotating whole image. In particular, the coordinates of each black pixel are rotated to save temporary storage and in this manner to decrease the calculation that would be required for a brute force implementation.

Bloomberg et al. [10] additionally utilize projection profiles to decide the text skew angle. Their algorithm contrasts from Postl's and Hull's algorithm in a manner that the images are down sampled before the projection profiles are ascertained to decrease computational expenses. The criterion function used to appraise the text skew angle is the change of the quantity of black pixels in a scan line.

Kanai and Bagdanov [11] display another text skew angle estimation algorithm based on projection profiles. The algorithm removes fiducial points and utilizes them as perspectives in the image by translating the most minimal resolution layer of the JBIG compressed image. The JBIG standard comprises of two strategies, a dynamic encoding strategy and a lossless compression technique for the least resolution layer. These points are projected along parallel lines into a collector matrix. The text skew angle is calculated as the angle of projection inside a pursuit interim that expands arrangement of the fiducial focuses. This algorithm recognizes a skew angle in the restricted range from $\pm$ 5 degrees to $\pm$ 45 degrees.

Papandreou and Gatos [12] utilize vertical projections for text skew detection with the standard capacity being the sum of squares of the projection components. This technique is guaranteed to be impervious to noise and image twisting and to work best for the dialects where most characters incorporate no less than one vertical line, which is valid for Latin-based dialects. In a later publication [13], Papandreou et al. report utilizing least bouncing box zones of joined horizontal and vertical projection profiles to decide record text skews. They assert that this approach is more impervious to noise and image twist, has no range limitations on text skews, and is appropriate for printed archives.

Li, Shen, and Sun [14] integrate projection profiles with wavelet disintegration. Document images are isolated into sub-images with the discrete wavelet change (DWT). The

matrix with the absolute values of the horizontal sub-band coefficients is rotated through a range of angles. A step size of 2 degrees is utilized to process an underlying gauge of the skew angle α. A better inquiry is then executed from α - 1 to α + 1 with a step of 0.5 degrees. The algorithm is assessed on an informational index with skews from 0 to $\pm$ 15 degrees.

Shivakumara et. al. [15] propose a document skew angle estimation approach in light of linear regression. They utilize linear regression formula keeping in mind the end goal to evaluate a skew angle for every text line section of a content document. A piece of the text line is removed utilizing static and dynamic thresholds from the projection profiles. This technique depends on the presumption that there is space between text lines. The technique loses precision for the documents having skew angle more prominent than 30 degrees and seems to work best for printed documents with separated lines.

Second class of algorithms utilize texture based ways to deal with assess document skew angles. Algorithms in this classification calculates discriminative features on pieces of text utilizing image filters to find pattern that are novel to the language or the script.

Chaudhury et. al [16] proposed utilizing a frequency domain portrayal of projection profiles of horizontal text lines. Busch et. al [17] exhibit a broad assessment of an expansive number of texture features, including projection profiles, Gabor and wavelet features and gray-level co-occurrence grids for distinguishing the content. This class of methodologies has the downsides of requiring extensive and adjusted homogeneous districts of text in one script, and of the features being referred to frequently being neither exceptionally discriminative nor reliable to compute within the sight of noisy or skewed text.

The third class contains algorithms which instrument associated components based techniques. These use shape and stroke attributes of individual associated components.

Hochberg et. al [18] proposed utilizing content specific layouts by clustering often occurring character or word shapes. Spitz et. al [19] developed shape codes that catch the concavities of characters, and utilize them to first order them as Latin-based or Han-based, and afterward inside those classifications utilizing different shape features.

Ma et. al [20] utilize Gabor-channels with a closest neighbor classifier to decide content and font-type at the word-level. A few hybrid variations of local and global approaches have likewise been recommended [21].

Kulyukin et. al [22] tried to localize NLs skewed upto 35-40 degrees either side from vertical. It utilizes edge detection, line detection, and corner detection. This technique is used to detect the NFT boundaries assuming that the image is not cropped and it is horizontally or vertically aligned. Once the boundaries are detected it is used for text chunking. The restriction of this algorithm is its inability to handle arbitrary text skews.

Zaman et. al [23] utilizes several iterations of 2DHWT for down-sampling image to identify horizontal, vertical and diagonal change of an image and store them in different n x n matrices. These matrices are utilized to compute the change set. Once the change set is obtained then convex hull algorithm is used to find minimum area rectangle which is then used to calculate the skew of text from the rotation of minimum area rectangle.

2.3 Deep Learning

Wang, Wu, Coates and Ng et. al [24] proposed to use sliding detector to discover lines in the image which helped in word segmentation and recognition. This was achieved using 2 convolutional layers with different filter values in a CNN. In this developed system, it was assumed that they have a lexicon which is a list of words. This was a substantial assumption as we can use previous information to oblige the search to only certain words in numerous applications.

Zhang, Zhao, LeCun et. al [25] used modular design to achieve optimization. The main module is convolutional module which calculates 1D convolution of the input. The model acknowledge a succession of encoded characters as input. The encoding is completed by endorsing an alphabet of size m for the input dialect, and after that quantize each character utilizing 1-of-m encoding. They developed 2 CNN both 9 layers deep consisting 6 convolutional layers and 3 fully-connected layers. Lastly, they used data augmentation to reduce the generalization error of deep learning network.

Joulin, Grave, Bojanowski, Mikolov et. al [26] tried to show that linear models with

a rank constraint and a fast loss approximation can train on a billion words within ten minutes, while achieving performance on par with the state-of-the-art.

Ciregan, Meier and Schmidhuber et. al [27] tries to prove that unsupervised pre-training of deep, hierarchical neural networks improves supervised pattern classification and using backpropagation on them improves the accuracy. DNNs fully unfold their potential when they are wide (many maps per layer) and deep (many layers).

Xue, Xing, Yang, Yu et. al [28] first tried to explore ways to increase text classification. They divided their approach into 2 phases. In first, they organized the hierarchy into flat categories, where they tried to find related categories of document. They prioritized categories. In second, they trained a classification model on a small set and then classified the input document. The architecture is utilized to improve the accuracy.

CHAPTER 3

IMAGE SEGMENTATION AND OCR

3.1 Introduction

Vision-based extraction of nutritional information from nutrition labels (NLs) accessible on most available packaged food products is the key for proactive nutrition management, since it enhances the client's capacity to take part in ceaseless nutritional information accumulation and analysis. Computer vision can be the key in the food selection process by bestowing consumers with real time text analysis of NLs, which will probably lock in consumers in proactive nutrition management [29]. An algorithm was built in past, a vision-based localization calculation for on a level plane or vertically adjusted NLs on for horizontally or vertically aligned NLs on smart phones [29]. The algorithm was subsequently changed to process not just adjusted NLs, yet in addition marginally skewed ones. A restriction of the algorithm was its powerlessness to handle self-assertive content skews [22]. New algorithm was proposed after modification in the older algorithm which was called TSAW(Text Skew Angle Wavelets) and was implemented in Java.

TSAW takes printed text images and down-samples them with a few repetitive times application of the 2D Haar Wavelet Transform (2D HWT). It first calculates the contour of the input image followed by computing the skew angle of the image and finding 4 corner co-ordinates. The skew angle is the angle of the image from vertical. This image is then chunked into different rows on the basis of horizontal line separators, as each row consists of different nutrition labels (NLs) with their daily value percent. These images were supposed to be sent to OCR (Tesseract) to get the text from the images. But the limitation here is that the OCR does not detect the whole word correctly, it does not work with rotated images, OCR is trained for converting image into text for font style "Arial" but US Food and Drug Administration (FDA) has standardized the font style as "Helvetica", for which

Tesseract is not trained. TSAW was designed in such a way that it had a great feature of correcting the text obtained as output from OCR. We calculate the distance of text from all the nutrition labels (NLs) stored in the form of Dictionaries using Levenshtein Distance algorithm.

Levenshtein Distance boosts the accuracy of the output. But still the other limitation makes the product less efficient. Apart from this, there are scenarios where even the Levenshtein Distance does not help us get the correct nutrition label. To improve it some more features were added to the algorithm.

In this chapter, we will try to explain all the modifications done to the algorithm TSAW to overcome the limitations. The original implementation of TSAW was done in Java and is openly available on git [30]. The new proposed changes were done in Java. To provide the reproducibility and truthfulness of the outcomes provided here, we have made our code publicly accessible on the git [31].

3.2 Image Rotation

The chunked images obtained from TSAW are skewed images which when passed to OCR, no OCR will be able to convert these images into text. The reason behind this is the skewed images and OCR works for vertical images or for images where text is horizontal. The output of the TSAW algorithm is shown in Fig. 3.1. All images are nothing but a 2D matrix where each index contains the pixel value of the image. The key to perform any operation on an image can be done by performing the operation on the matrix or the pixel values and these changes can be seen on the image. So to get the best out of any OCR we need to give the input images in the form OCR understands the best. The best possible solution for this is to use the skewed angle obtained from TSAW algorithm to rotate all the images to vertical.

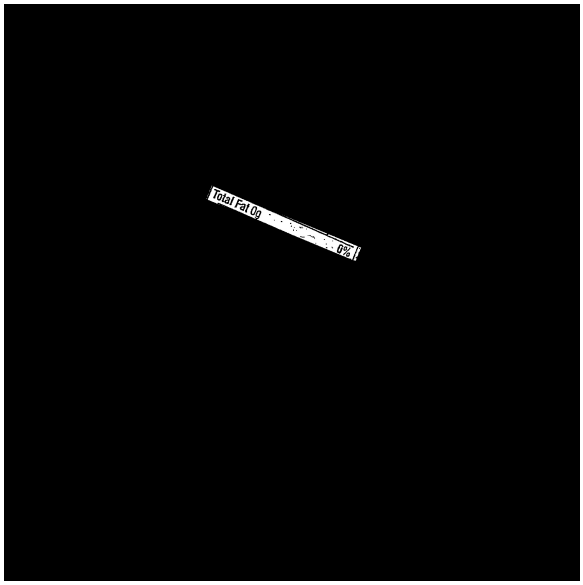

Fig. 3.1: Skew chunked image

The rotation of any 2D object is done about a certain point. We considered center of the image as the point of rotation. We take the skew angle calculated in TSAW and subtract it from 90 and use this value as the angle for image rotation. To rotate the image we will calculate the affine matrix. The function that calculates affine matrix is:

$$\begin{bmatrix} \alpha & \beta & (1-\alpha)\cdot center\cdot x - \beta\cdot center\cdot y \\ -\beta & \alpha & \beta\cdot center\cdot x + (1-\alpha)\cdot center\cdot y \end{bmatrix}$$

where,

$$\begin{aligned} \alpha &= scale\cdot\cos angle \\ \beta &= scale\cdot\sin angle \end{aligned} \tag{3.1}$$

After getting affine matrix, we apply affine transformation on the image. Affine transformation is a direct mapping technique that conserve points, straight lines, and planes. The affine transformation procedure is ordinarily used to counteract for geometric contortions

or distortions that happen with non-ideal camera angles.

$$dst(x, y) = src(M11 \cdot x + M12 \cdot y + M13, M21 \cdot x + M22 \cdot y + M23) \tag{3.2}$$

The rotated matrix of the original image is then saved on file system in the form of image. The output of the rotated image can be seen in Fig. 3.2.

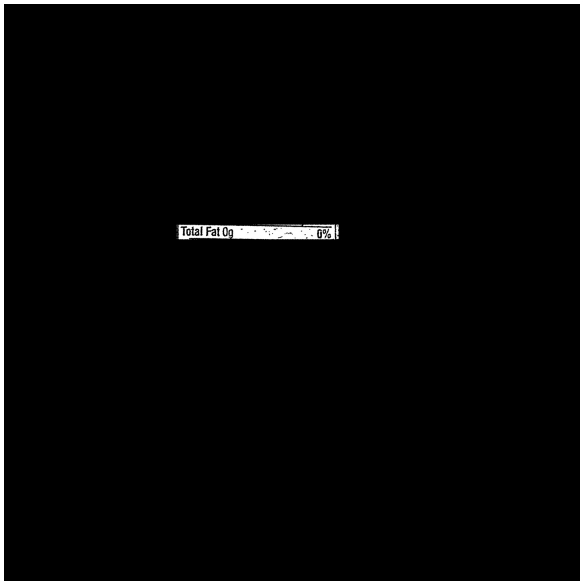

Fig. 3.2: Rotated chunked image

3.3 Rotated Image to OCR

As the images we passed to the OCR were not rotated, the accuracy of OCR was very less. In the last section, we have tried to find a way to rotate all the chunked images to make them vertical so that we can pass them to OCR as it is trained only to convert images to text which are vertical. We now pass these rotated images to OCR one by one and save all the text obtained in the text file.

We analyzed the output text file of OCR and found the words obtained from OCR were more close to NLs and more accurate. But we found that there were still lots of words Tesseract was unable to convert for example, the OCR was converting the "g" representing

grams in the nutrition table as "9" and there were other similar errors. This inaccuracy was because of font style difference. As the font styles OCR or Tesseract are trained for does not contain the font style used for creating nutrition table. This was restricting us to increase the accuracy.

3.4 Training Tesseract

After analyzing the output we observed that the rotated images passed to OCR improved the accuracy of the word prediction but, still it had issues of not identifying few characters as discussed in previous section. This was because of the font style training of OCR. The OCR used here is Tesseract which is the best OCR to convert images to text. Tesseract is trained for various font styles but the FDA standards use Helvetica as the font style which is not present in Tesseract.

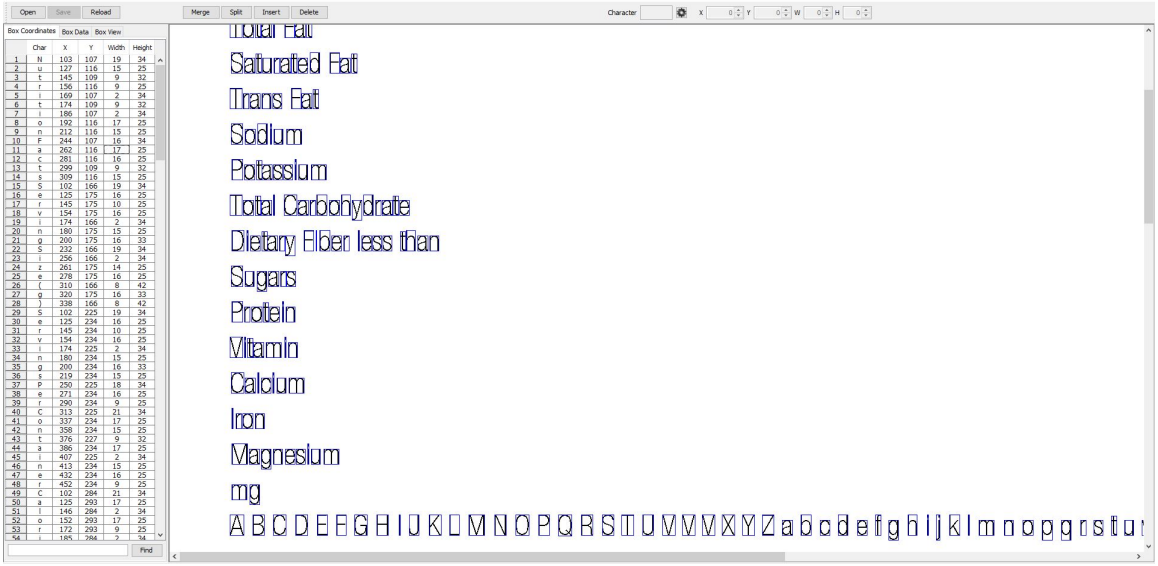

Fig. 3.3: The UI of jTessBoxEditor

The best way to overcome this restriction was to train Tesseract for the specific font style and font size. After doing some research on the ways to train Teseract, the best tool to train Tesseract can be achieved using a Java based tool called "jTessBoxEditor". It is shown in Fig. 3.3. It is a box editor that can be used for full automation training of

Tesseract. It can read common image format which includes multi page .tiff file as input and then click the "Run" button. It also allows us to customize the imported text from the files by providing the dimensions of the letters. Once the training is complete, save the trained state of tesseract. After saving this state, whenever we try to access Tesseract, we will have access to the new trained font style.

3.5 OCR Output vs Trained OCR

Once the Tesseract is trained and easily accessible, we can use it to convert the image passed as input to text for the trained font with more accuracy. We observed that the output obtained from trained OCR was able to predict more words and characters or words more close to the actual nutrition labels. But still it was not sufficient to achieve good accuracy and cannot be relied on too much because if in future FDA change the fonts again it may be too much efforts.

Fig. 3.4: The input image

Fig. 3.5: Output of image

The image given as input is shown in Fig. 3.4. This image is processed and chunked into different rows using the lines in the image. When these chunked images after rotation

are passed to OCR, the OCR converts the image to text. The output obtained after applying Levenshtein Distance on the output of OCR output is saved in the form of a text file shown in Fig. 3.5.

3.6 Improvement to Text Chunking

Apart from all the changes done earlier, there were still some scenarios in which the image chunking wasn't working which was restricting us to create a dataset of images on which we can test. The reason behind image segmentation not working includes processing and chunking vertical image or we can say images with angle 90 degree. To improve image chunking, we modified the existing code by adding the logic of calculating the coordinates of the image in vertical image scenario because the 4 corner coordinates of image are the base of calculating the line segments and all other calculations required for the image segmentation. The algorithm behind calculating 4 corners of image can be seen in Algo. 4.1.

Algorithm 3.1 Cropping a Nutrition Label from an image

Input:
Calculate the 4 corners of vertical image
Output:
4 coordinates of the vertical image
Begin
Find the list of points in the image whose pixel value
Find minimum area rectangle from list of points
Using the minimum area rectangle we can find the angle of rectangle
Find the vertices of minimum area rectangle
Calculate the height and width of minimum area rectangle
End

We also analyzed that the image had some noise which was affecting the image segmentation and OCR output. So, we made some calculation changes to avoid the black borders using the comparison of black pixel count and line segment length. Modified the algorithm little bit to reduce the noise content in the images. Noise reduction was required because each time when we read and write the image, some noise is introduced inside the image.

CHAPTER 4
IMAGE CLASSIFICATION

4.1 Introduction

We tried to bring lots of improvements to OCR and image processing algorithm to increase the accuracy, but after some time we analyzed that the improvement in the efficiency to classify the nutrition labels (NL) got stagnant. No further improvements were adding anything new to the implementation.

After some research we came to a conclusion to encapsulate image processing with Machine Learning (ML) algorithms to get better accuracy. Innovations on machine learning has powered various sectors of modern society: starting from web search to content filtering on social networks to suggestions on e-commerce websites, and it is expanding towards consumer products like cameras and smartphones.

4.2 Supervised Learning

The most widely recognized type of machine learning is supervised learning. Amid training, the machine is demonstrated with a picture and delivers an output as a vector of scores, one for every classification. We calculate an objective function that evaluates the error (or distance) between the output scores and the coveted example of scores. The machine at that point alters its interior customizable parameters to lower this error. These customizable parameters, mostly called weights, are real numbers that can be viewed as 'handles' that characterize the inputoutput function of the machine.

4.3 Deep Learning

Machine-learning frameworks are utilized to recognize objects in images, transform speech into text, match news items, find posts or products of users interests, and select significant results of search [32]. Conventional machine-learning techniques were restricted in

their capacity to process normal information in their raw shape. Since a long time, designing a pattern-recognition or machine-learning framework required a cautious engineering and significant domain proficiency to design a feature extractor that metamorphose raw data (for example pixel value) of an image into a reasonable inward portrayal or feature vector from which the learning subsystem, often a classifier, could identify or classify patterns in the info. To overcome this limitation, we can make use of a class of systems called deep learning.

Representation learning is an arrangement of techniques that enables a machine to be sustained with raw information and to naturally discover the representations required for detection or classification of objects. Deep-learning techniques are representation learning techniques with different levels of portrayal, acquired by making straightforward however not-changing modules that each change the representation at one level into a representation at a higher, marginally more dynamic level.

The key part of deep learning is that these layers of features are not developed by human designers: they are gained from information utilizing a broadly useful learning strategy. Deep learning is making real advances in tackling issues that have opposed the best endeavors of the artificial intelligence group for a long time. It has ended up being great at finding multifaceted structures in high-dimensional information and is therefore appropriate to be applied to numerous domains of science, business and government.

4.4 Back Propagation to Train Multilayer Architecture

From the preliminary times of pattern recognition, researchers endeavor to achieve has been to supplant hand-designed features with trainable multilayer networks. Multilayer architectures can be trained by straightforward stochastic gradient descent. For whatever length of time that the modules are generally smooth functions of their inputs furthermore, their internal weights, one can process gradients utilizing the backpropagation technique.

The backpropagation strategy to process the gradient of an objective function regarding the weights of a multilayer stack of modules is simply a practical use of the chain rule for derivatives. The simple form of backpropogation algorithm for weight updation cosists of 3

steps with the following equations:

1. Feed forward training instances

$$x_i = f_i(W_i \cdot x_i) \tag{4.1}$$

where,

x_i is input vextor

f_i is activation function

W_i is weight

2. Calculate the error for layer l

$$\delta^l = ((w^{l+1})^T \cdot {}^{l+1}) \odot \sigma'(z^l) \tag{4.2}$$

where,

$(w^{l+1})^T$ is transpose of weight matrix for l+1 layer

$^{l+1}$ is the error at layer l+1 $\odot\sigma'(z^l)$ is Hadamard product

3. update the weights

$$W_i = W_i - \alpha \cdot w_i \tag{4.3}$$

where, α is learning rate

The key knowledge is that the subordinate (or gradient) of the goal regarding the contribution of a module can be calculated by working in reverse from the gradient regarding the output of that module.

4.5 Convolution Neural Network

ConvNets are intended to process information that come as numerous arrays. There are four key concept behind ConvNets that exploit the properties of regular signals: local connections, shared weights, pooling and the utilization of numerous layers.

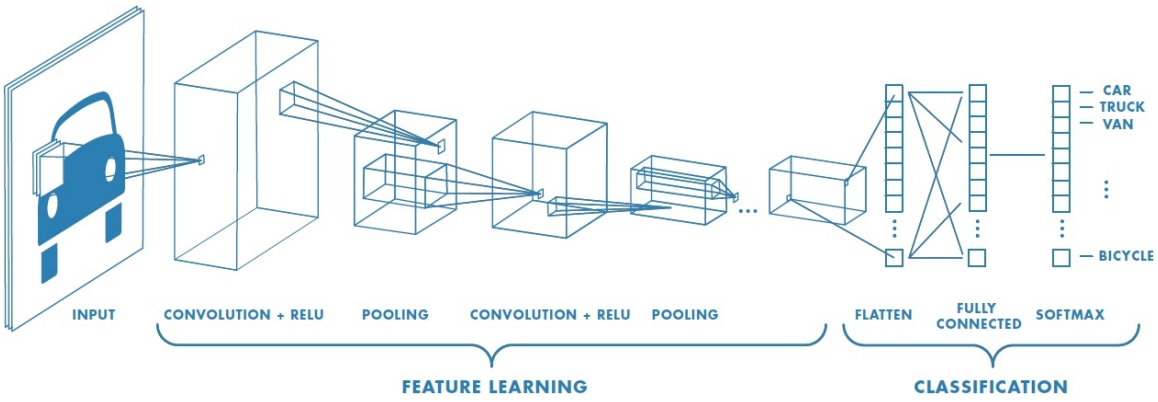

Fig. 4.1: Example of Convolutional Neural Network

The architecture of a simple CNN is organized as a arrangement of stages as shown in the form of an example in Fig. 4.1 [33].The part of the convolutional layer is to identify nearby conjunctions of features from the previous layer, the part of the pooling layer is to consolidate semantically comparable features into one. Deep neural networks exploit the property that numerous characteristic signs are compositional hierarchies of command, in which more elevated amount of features are acquired by forming lower-level ones. In images, local combinations of edges shape motifs, motifs gather into parts, and parts shape objects.

4.6 Image Classification

To start, we embrace the CNN representation to handle the issue of image classification of items and scenes. The framework ought to dole out (possibly various) semantic labels to an image. Keep in mind as opposed to object detection, object image classification requires no limitation of the objects. The CNN representation has been optimized for the object image classification.

4.7 Dataset and Labeling

The data has been collected by going to grocery store and taking the snaps of nutrition tables printed on packaged food items using smartphone samsung s3 which has 8MP rear camera. These images are then copied on the computer and passed to the modified java implementation of TSAW. The rotated output of the TSAW is shown in previous chapter.

The data obtained from TSAW was manually cropped and labeled into 24 classes for training data.The nutrition label images obtained in the previous chapters have the % value in images, which needs to be removed. So these train images were manually cropped and put in the labeled set. This manual cropping and labeling is done only for the training dataset. The train image can be seen in Fig. 4.2.

Fig. 4.2: Cropped training image

Similar to training dataset we have to create validation dataset but instead of cropping validation images manually, we wrote a python code for segmenting the image into nutrition label part and percent part with the use of black pixel mean. The algorithm can been in Alg. 4.1.

The output of the cropping image algorithm can be seen in Fig. 4.3 and 4.4.

Fig. 4.3: The validation image

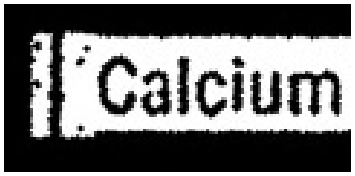

Fig. 4.4: The validation image

Algorithm 4.1 Cropping image to obtain only Nutrition Labels

Input:

All images whose extension is *.jpg

Output:

All segmented images taken as input is saved as *.jpg

Begin

```
Apply Dilate morphological operation on image of kernel size 5
Apply Erode morphological operation on image of kernel size 2
For each 3 columns in all columns of image
  For all rows in image
    if pixel value = 255
      count number of black pixel
  Calculate mean of 3 count of black pixels for 3 columns at a time
Find drops and spikes in the means
For each 3 values check if the difference ¡ 3
  get the value of drop
```

End

After all these operations we got the dataset for CNN. The dataset used for training and validation of the CNN can be seen in Table 4.1:

Table 4.1: Dataset Information

No. of Train images	1266
No. of Validation images	128
No. of Classes	24

4.8 Tuning

There is no specific and decided architecture of any neural network including CNN. Everyone has to work on designing and finding the best model and architecture of the CNN. This can be achieved by altering the hyper-parameters of the CNN. Hyper-parameters can be anything starting from number of filters to activation function or changing the layers of CNN in architecture to number of epochs. We performed changing of most of these

parameters to improve the accuracy of the CNN to its maximum.

4.9 Design CNN

CNN has proven to be very efficient in image classification and recognition. There is a restriction in engineering artificial neural network or any traditional machine learning algorithms to have domain knowledge but CNN surpass this limitation as it automatically adjust itself for different domains.

In our scenario, we have used an architecture where we have used 2 convolution layers followed by 1 pooling/sub-sampling layer. The pooling layer is then followed by fully connected layer. The complete architecture of the CNN used in our scenario is shown in Fig. 4.5

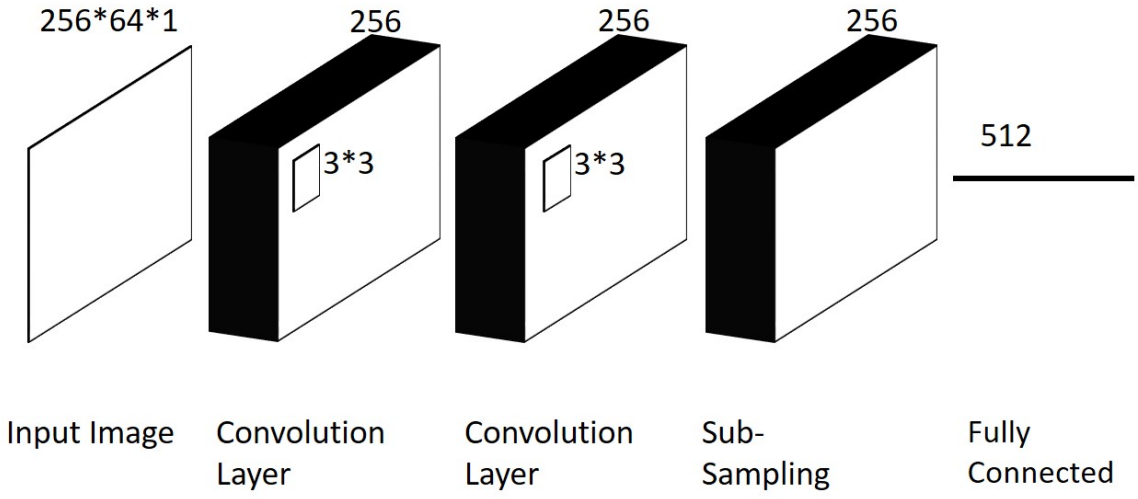

Fig. 4.5: Example Convolutional Neural Network

The layers in any deep learning algorithms are connected and takes the output of previous layer as input. When a layer in Deep Learning perform computation on the matrix passed as input gives an output for the next layer. Each filter in convolution layer is duplicated across the whole layer. This replicated filters form feature maps. Each position brings about an activation of the neuron and the output is gathered in the feature map.

Feature map can also be considered as an encoding where feature stands for what is available in the image while map encodes where this feature is present in the image. The feature map of my CNN architecture is shown in Fig. 4.6

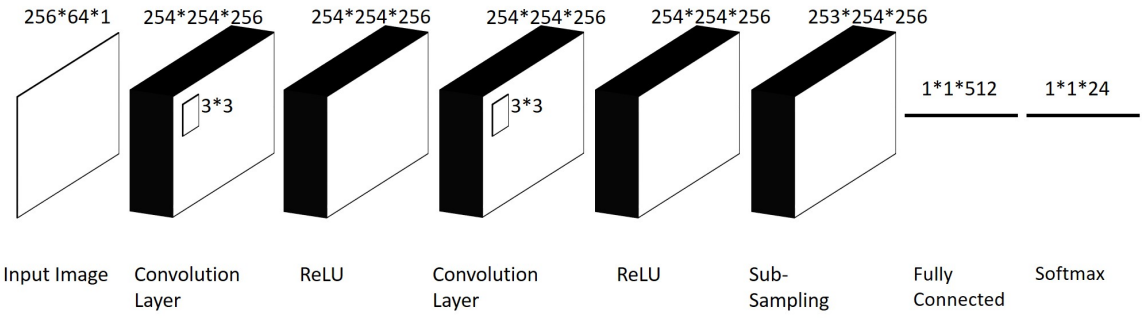

Fig. 4.6: Feature map of the CNN architecture

CHAPTER 5
EXPERIMENTS AND RESULTS

5.1 Introduction

In previous chapter, we discussed the architecture and feature map of CNN. We also saw how CNN is a good option for image classification. While engineering any machine learning or deep learning algorithms, there is no single standalone solution to problems. The architectures of any algorithm vary in different ways for different scenarios. The architecture of any ML algorithm designed and optimized for one type of input may or may not work for other types of input.

To design or optimize any ML algorithm we have to tune it for specific scenarios by changing different parameters or components to see in which scenario it performs best or gives the highest accuracy.

5.2 CNN Hyper-parameters

We have discussed about tuning a CNN in previous chapter and previous section. Hyper parameters are the parameters passed to any layers in model which can be anything like loss, activation function, filters dropout etc. I initially designed my CNN with 2 convolution layers followed by 1 pooling layer then again 2 convolution layers followed by pooling layer and finally a fully connected layer. The filters in first block of convolution is 256 and 512 respectively. The number of filters in fully connected layer is kept at 512. The train set is divided into a batch size of 32. The Kernel size is a window of 3x3 and then it is slided by 1 pixel. Sliding the kernel window is called stride. The kernel size in pooling layer is 2x2. Each layer block had ReLU as activation function. The value of Dropout function was 0.25. We used Categorical Cross Entropy as the loss function and the number of epochs were 20. But this was the initial state of the CNN. We changed various parameters and

analyzed the accuracy and loss of the CNN. If we got higher accuracy and lower loss then we moved forward with keeping those changes otherwise we reverted the model to previous state. Below we have discussed few hyper parameters and how changing them affected our accuracy.

5.2.1 Optimizers

A loss function or cost function is a capacity that maps a values of at least one variable onto a real number instinctively representing to some "cost" related with the event. A loss function deliberates the nature of a specific arrangement of parameters based on how well the instigated scores fit with the ground truth labels in the training data. The objective of optimizer is to discover weight W that reduces the loss function. There are different types of optimizers that can be used in deep learning.

- SGD - SGD stands for Stochastic Gradient Descent, update the parameters for every training set.
- RMSprop - It is an adaptive learning method.
- Adagrad - This is an algorithm for gradient optimization. It adjusts the learning rate to the parameters.
- Adadelta - It is an enhancement to Adagrad that tries to lower learning rate.
- Adam - It stands for Adaptive Moment Estimation, which calculates adaptive learning rate for each and every parameter. It can be seen as the integration of RMSprop and momentum.
- Adamax - It is a variation of Adam built on infinity norm.
- Nadam - It stands for Nesterov Adam optimizer. It is an mixture of Adam and NAG.

The output of the CNN for optimizer is shown in table 5.1.

Table 5.1: Experiment results on changing Optimizers

	Epoch	**Training Loss**	**Validation Loss**	**Training Accuracy**	**Validation Accuracy**
SGD	20	2.8793	3.1366	16.28	15.64
RMSprop	20	2.0867	2.57	33.30	31.24
Adagrad	20	13.7311	14.5645	14.81	15.64
adadelta	20	2.2812	2.7533	31.12	30.04
Adam	20	2.0868	2.1265	33.87	32.03
Adamax	20	2.0856	2.9450	34.85	31.47
Nadam	20	2.8532	3.9982	13.03	12.31

After analyzing these experiments, we can interpret that Adam and Adamax outperform in accuracy than other optimizers.

5.2.2 Epoch

Epoch is very much similar to iterations, but can be better defined as one forward and backward pass of all the training examples available. It can also be considered as the number of times the model sees training set. We analyzed that 20 epochs is very less so we tried different epochs like 60 for optimizer Adam as the network gave best output in Adam. The output of the CNN for Epoch is shown in table 5.2

The output of the CNN for increasing epochs for Adam optimizer is shown in table 5.2.

Table 5.2: Experiment results on changing Epochs

Number of Epoch	**Training Loss**	**Validation Loss**	**Training Accuracy**	**Validation Accuracy**
60	1.1860	1.5082	60.20	58.27

After this experiment our decision to increase epoch was effective as the accuracy increased.

5.2.3 Dropout

In previous section we found that increasing epoch helped us achieve higher accuracy so we will move ahead with this new architecture. Dropout layers have a particular purpose in neural systems. This layer "drops out" an arbitrary set of activations in that layer by making them zero. It ensures that the system is not getting excessively "fitted" to the training data and accordingly reduces the overfitting issue. The output of the CNN for Dropout is shown in table 5.3.

Table 5.3: Experiment results on reducing Dropout

Dropout value	**Training Loss**	**Validation Loss**	**Training Accuracy**	**Validation Accuracy**
0.2	1.0825	1.4666	64.79	61.73

In this experiment we observed that the accuracy of the CNN increased by reducing the value of Dropout.

5.2.4 Reduced Validate Dataset

In previous section of experiment, we saw that the accuracy increased with the architecture change so we will proceed with those changes, but reducing validation data can

be effective as the validation data has noisy data. There were lots of images which in the validation dataset which were very noisy due to which they were not properly chunked by the python code written to chunk them. The algorithm is discussed in previous chapter. So, we removed all such images. The output of the CNN for reduced validate set is shown in table 5.4.

Table 5.4: Experiment results on reducing validation dataset

	Training Loss	**Validation Loss**	**Training Accuracy**	**Validation Accuracy**
Reduced Validation dataset	0.9253	1.1567	69.38	68.56
Epoch 70, Reduced Validation dataset	0.9103	1.1648	69.88	68.23
Epoch 80, Reduced Validation dataset	0.8979	0.9119	70.15	69.64
Epoch 120, Reduced Validation dataset	0.7128	0.7896	76.85	74.50
Epoch 180, Reduced Validation dataset	0.5176	.0.8359	83.04	82.84

In this section we see that the accuracy increased by removing noisy data.

5.2.5 Convolution Layer, Layer Block

The architecture changes done to CNN till now has helped us get higher accuracy. In previous section, we discussed the architecture having 2 convolution layers and 1 pooling layer. We considered it as a block. We added 1 extra convolution layer in the block and tried adding and removing the whole layer block. The output of the CNN for reduced validate set is shown in table 5.5.

Table 5.5: Experiment results on changing Convolution layer and Convolution Block

	Training Loss	**Validation Loss**	**Training Accuracy**	**Validation Accuracy**
Adding convolution layer	1.5300	1.6295	49.85	46.73
Adding layer block	15.92	17.35	6.48	4.66
Removing layer block	0.9680	1.2271	68.02	66.94
Epoch 70, Removing layer block	1.3246	1.7945	57.85	56.81

In this experiment, we can observe that removing layer block also gave a good accuracy.

5.2.6 Adamax Optimizer

In previous section, we discussed about Adamax optimizer. We also observed that Adamax gave good accuracy same as Adam optimizer. So we tried performing some similar experiments performed on Adam with Adamax optimizer function. The output of the CNN for reduced validate set is shown in table 5.6.

Table 5.6: Experiment results on changing epoch for Adamax optimizer

	Training Loss	**Validation Loss**	**Training Accuracy**	**Validation Accuracy**
60 Epoch with decreased layer block	1.0745	1.2212	66.51	64.97
70 Epoch with decreased layer block	1.0521	1.153	66.27	63.45

In these experiments we can infer that although Adamax gave good accuracy but its not higher than Adam.

5.2.7 Filters, Kernel

In this section we tried to make changes to filters value and kernel size to see if we can achieve higher accuracy. The first layer of any CNN can only be a Convolutional Layer. The input to this convolution layer will be an array containing pixel values of the image. A filter is represented by a vector of weights with which we convolve the input. Filters are also known as neuron. A vital note is that the depth of the filter must be the same as the depth of the input. The output of the CNN for reduced validate set is shown in table 5.7.

Table 5.7: Experiment results on changing Filters

	Training Loss	**Validation Loss**	**Training Accuracy**	**Validation Accuracy**
Decreasing Filter to 128	0.9852	1.2903	66.99	63.14

Decreasing filters value has given good results but it decreased the accuracy by a small margin.

Every convolution operation has a kernel which can be a matrix of any size lesser than the original image in width and height. Every kernel is valuable for a specific work, for example, sharpening, obscuring, edge detection, and many more. The output of the CNN for reduced validate set is shown in table 5.8.

Table 5.8: Experiment results on changing Kernel size

	Training Loss	**Validation Loss**	**Training Accuracy**	**Validation Accuracy**
Increasing Kernel to 5x5	12.98	13.8903	13.99	10.14
Decreasing Kernel to 2x2	1.0888	1.4739	63.87	59.94

In this experiment, increasing the kernel size reduced the efficiency by a huge factor while decreasing kernel size has given good results but it decreased the accuracy by a small margin.

CHAPTER 6

CONCLUSION

A legitimate comprehension of nutrition labels (NLs) is fundamental to guarantee eating a healthy, unprejudiced eating routine. These labels give data on the measures of 13 fundamental nutrients and calories in a measure of food, alongside a % Daily Value pointer to enable individuals to make informed decision over food choices. This information is introduced as a standardized table. Commonality with the terms of the NLs enables a buyer to make a superior choice while looking for packaged food items and contrasting one item to another. Nonetheless, numerous customers think that its hard to interpret the nutritional information on items and feel less motivated to monitor their nutrient utilization. One approach to enhance the cognizance and maintenance of nutritional data by consumers is to utilize computer vision algorithms that can keep running on a cell phone. This incorporates filtering the Nutrition Label itself and separating the nutritional information from it.

The primary challenge of this dissertation is to classify nutrition labels and the values in the input image of nutrition table. But before solving that we have to make lots of changes to the TSAW algorithm used to find out the skew angle of the image. Because the chunked image obtained after TSAW are to be passed in OCR which was considered to be the ground truth for the image classification protocol. But the limitations of TSAW were analyzed and were improved by first applying the image rotation on the chunked images of nutrition labels because OCR can transform horizontal images to text in a very efficient way. After passing the rotated images to OCR we found the reason for lower accuracy was also the font style for which OCR is trained is not the same as the font style used in nutrition table. To come over this limitation we should train the OCR for the specific font style. For this we used a tool called jTessBoxEditor where we can train Tesseract OCR for any font style and font size and then use the output of this tool for OCR. After making these changes we observed that the TSAW was unable to handle the vertical images passed as

input. We made couple of changes for calculating the coordinates of the image for vertical images. We also made some changes to remove borders from images because they act as noise if we pass them to OCR or any machine learning algorithm.

To deal with the main challenge of image classification we can use machine learning algorithms. But a special class of algorithms was developed called Deep Learning which has proven to be very efficient in various sectors. Deep learning has various algorithms which are specifically designed for specific scenarios. In our case we have to classify images and CNN has been specially designed keeping in mind classification and recognitions in images, audio and video. Our scenario was a supervised learning because we had labeled training set and validation set. The biggest thing to get the most optimum neural network is tuning them by changing the hyper parameters. The architecture used by us of CNN was 2 layer blocks each having 2 convolution layers with 1 pooling layer. These 2 layer blocks were followed by fully connected layer.

6.1 Lower Accuracy Reason

The highest accuracy we were able to achieve was ≈ 69 % which was greater than the highest accuracy achieved using TSAW and OCR. But still this accuracy is not enough because CNN are well known for very efficient and higher accuracy between 80-99%. The reason behind low accuracy are:

6.1.1 Dataset

The dataset we use consists of training and validation dataset.

- Input image dimension - The input image we are using has little variations in the height and width of the image.

- Small Dataset - Any machine learning algorithm requires thousands of images to train and have higher predictions. Even for CIFAR-10 dataset they have around 60000 images in 10 classes with 6000 images in each class. In our scenario we have around 24 classes for classification. A lot of times it is suggested that each class should have

at-least 1000 images in each class. This could be a very big reason of lower accuracy as we only have 1266 images for training.

- Noise - If the training dataset has noisy data or mislabeled data, it can be a reason behind lower accuracy. In our scenario, we do not have any mislabeled data but the train dataset has too much noise or unwanted black pixels added to images. The images in training dataset is not very clear.

- Batch Size - It is defined as number of samples used in the network at a time. Higher batch size reduces the generalization of the CNN which affects the accuracy of the model. The batch size used in our scenario is 32. It is a huge possibility that by decreasing the batch size to 16 or 8 to get higher accuracy.

6.1.2 Image Augmentation

Image Augmentation can be defined as a technique of manipulating the images in training dataset to create multiple interpolated images. This provides more images to train and also provides wide varieties of lightning, coloring and skewed images for CNN to train. We are using image augmentation for our cnn. But not proper augmentation on the train set may reduce the accuracy. We may need to change the rotation angle, blurring value etc.

6.1.3 High Loss

The loss is calculated by taking the negative ln of 10% of number of classes used. In our scenario we have 24 classes. So, it will be -ln(0.24). The loss calculated is 1.427.

Another big factor in lesser accuracy could be "Pattern too complex to learn". It is a possibility because we have 24 classes of classification and images have some noise which makes the training set more complicated. The patterns have different curves and edges which brings variety of data and makes training complex. If the image resolution is big then 2 layers can be not sufficient to train network for big image.

6.1.4 Hyper-Parameters

Hyper parameters are the values used for tuning a CNN.

- Number of epochs - In our scenario we are using epochs as 60 and we did experiments by increasing the epochs to 70 and 80, but this is very less because sometimes MNIST gives 99% accuracy at epoch 300 where data is 32·32 resolution image so we may need to try higher epochs.

- Learning rate - Low learning rate cause the model to converge slowly, on the other hand, high learning rate decreases loss but not able to find solution. The learning rate Adam optimizer has is 0.001 but Adamax has learning rate of 0.002 so we can try by making a custom optimizer.

6.2 Future Work

Our future work mainly focuses on collecting more training and validation images as we had only 1266 train images and any machine learning algorithms gives higher accuracy on tens of thousands of images. We will also focus on tuning the CNN more and more by changing other parameters. Then we can create a user interface for the smart phones, android and ios both, so that users can use it for keeping track of their nutrition intake. This app will also be able to give suggestions to users on the basis of their health. But CNN require high computation so we can deploy it on a GPU based computation server.

REFERENCES

[1] U. S. D. of Agriculture, "Economic research service data." [Online]. Available: https://www.ers.usda.gov/amber-waves/2005/november/us-food-consumption-up-16-percent-since-1970/

[2] W. H. Organization, "Annual world health statistics." [Online]. Available: www.who.int

[3] N. Labeling and E. A. of 1990. [Online]. Available: http://en.wikipedia.org/wiki/Nutrition_Labeling_and_Education_Act_of_1990

[4] F. L. to Advance Better Education for Life. [Online]. Available: www.flabel.org/en

[5] D. H. S. Sinclair and S. Goodman, "Sociodemographic differences in the comprehension of nutritional labels on food products," *Journal of Nutr. Educ. Behav.*, vol. 45(6), pp. 767–72, 2013.

[6] R. J. D.J. Graham, "Location, location, location: eye tracking evidence that consumers preferentially view prominently positioned nutrition information," *J. Am. Diet. Assoc*, vol. 111, p. 17041711, 2011.

[7] W. Postl, "Detection of linear oblique structures and skew scan in digitized documents," in *In Proc. of International Conference on Pattern Recognition*, 1986, pp. 687–689.

[8] ——, "Method for automatic correction of character skew in the acquisition of a text original in the form of digital scan results," Feb. 2 1988, uS Patent 4,723,297.

[9] J. Hull, "Document image skew detection: survey and annotated bibliography.ws," *In J.J. Hull, S.L. Taylor (eds.), Document Analysis Systems II, World Scientific Publishing Co.*, pp. 40–64, 1997.

[10] D. S. Bloomberg, G. E. Kopec, and L. Dasari, "Measuring document image skew and orientation," in *Document Recognition II*, vol. 2422. International Society for Optics and Photonics, 1995, pp. 302–317.

[11] J. Kanai and A. Bagdanov, "Projection profile based skew estimation algorithm for jbig compressed images," *International Journal on Document Analysis and Recognition*, pp. 43–51, 1998.

[12] A. Papandreou and B. Gatos, "A novel skew detection technique based on vertical projections," *In Proc. of International Conference on Document Analysis and Recognition (ICDAR)*, pp. 384–388, Sept 2011.

[13] A. Papandreou, S. P. B. Gatos, and I. Gerardis, "Efficient skew detection of printed document images based on novel combination of enhanced profiles," *Int. J. Doc. Anal. Recognit*, pp. 433–454, 2014.

[14] Q. S. S.T. Li and J. Sun, "Skew detection using wavelet decomposition and projection profile analysis," *Pattern Recognition Letters*, pp. 555–562, 2007.

[15] D. G. P. Shivakumara, G. Hemantha Kumar and P. Nagabhushan, "Skew estimation of binary document images using static and dynamic thresholds useful for document image mosaicing," *In Proc. of National Workshop on IT Services and Applications (WITSA 2003)*, pp. 51–55, Feb 2003.

[16] S. Chaudhury and R. Sheth, "Trainable script identification strategies for indian languages," *Proc. 5th IEEE Intl. Conf. on Document Analysis and Recognition (ICDAR)*, pp. 657–680, 1999.

[17] W. B. A. Busch and S. Sridharan, "Texture for script identification," *IEEE Trans. Pattern Analysis and Machine Intelligence (PAMI)*, vol. 27(11), pp. 1720–1732, 2005.

[18] T. T. J. Hochberg, P. Kelly and L. Kerns, "Automatic script identification from document images using cluster-based templates," *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, pp. 176–181, 1997.

[19] A. Spitz, "Determination of the script and language content of document images," *IEEE Trans. Pattern Analysis and Machine Intelligence (PAMI)*, pp. 235–245, 1997.

[20] H. Ma and D. Doermann, "Gabor filter based multi-class classifier for scanned document images," *Proc. 7th IEEE Intl. Conf. on Document Analysis and Recognition (ICDAR)*, pp. 968–972, 2003.

[21] Y. L. L.J Zhou and C. Tan, "Bangla/english script identification based on analysis of connected component profiles," *7th IAPR Workshop on Document Analysis Systems (DAS)*, pp. 243–254, 2006.

[22] V. Kulyukin and C. Blay, "An algorithm for mobile vision-based localization of skewed nutrition labels that maximizes specificity," in *Emerging Trends in Image Processing, Computer Vision and Pattern Recognition*. Elsevier, 2015, pp. 277–293.

[23] T. Zaman and V. Kulyukin, "Text skew angle detection in vision-based scanning of nutrition labels," in *Proceedings of the International Conference on Image Processing, Computer Vision, and Pattern Recognition (IPCV)*. The Steering Committee of The World Congress in Computer Science, Computer Engineering and Applied Computing (WorldComp), 2015, p. 139.

[24] T. Wang, D. J. Wu, A. Coates, and A. Y. Ng, "End-to-end text recognition with convolutional neural networks," in *Pattern Recognition (ICPR), 2012 21st International Conference on*. IEEE, 2012, pp. 3304–3308.

[25] X. Zhang, J. Zhao, and Y. LeCun, "Character-level convolutional networks for text classification," in *Advances in neural information processing systems*, 2015, pp. 649–657.

[26] A. Joulin, E. Grave, P. Bojanowski, and T. Mikolov, "Bag of tricks for efficient text classification," *arXiv preprint arXiv:1607.01759*, 2016.

[27] D. Ciregan, U. Meier, and J. Schmidhuber, "Multi-column deep neural networks for image classification," in *Computer vision and pattern recognition (CVPR), 2012 IEEE conference on*. IEEE, 2012, pp. 3642–3649.

[28] G.-R. Xue, D. Xing, Q. Yang, and Y. Yu, "Deep classification in large-scale text hierarchies," in *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 2008, pp. 619–626.

[29] V. Kulyukin, T. Zaman, and S. Andhavarapu, "Effective use of nutrition labels on smartphones," in *15th Int. Conf. Internet Computing and Big Data*, 2014, pp. 21–24.

[30] "Java source code of the tsaw algorithm:." [Online]. Available: https://github.com/tanwirzaman/HaarTextSkewDetection

[31] A. Khasgiwala, "Word recognition in nutrition labels with convolutional neural network." [Online]. Available: https://github.com/anujkhasgiwala/Word-Recognition-in-Nutrition-Labels-with-Convolutional-Neural-Network

[32] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, p. 436, 2015.

[33] MathWorks, "Convolutional neural network." [Online]. Available: https://www.mathworks.com/content/mathworks/www/en/discovery/convolutional-neural-network/jcr:content/mainParsys/image_copy.adapt.full.high.jpg/1492406018870.jp