REPUBLIC OF TURKEY

ALTINBAŞ UNIVERSITY

Institute of Graduate Studies

Electrical and Computer Engineering

COMPUTER VISION AND DEEP LEARNING FOR AUTOMATIC DETECTION AND LOCALIZATION

Zinah Hayder Hammoodi ALHUSSEIN

Master`s Thesis

Supervisor

Prof. Dr. Galip CANSEVER

Istanbul, 2022

COMPUTER VISION AND DEEP LEARNING FOR AUTOMATIC DETECTION AND LOCALIZATION

Zinah Hayder Hammoodi ALHUSSEIN

Electrical and Computer Engineering

Master`s Thesis

ALTINBAŞ UNIVERSITY

2022

The thesis titled COMPUTER VISION AND DEEP LEARNING FOR AUTOMATIC DETECTION AND LOCALIZATION prepared by ZİNAH HAYDER HAMMOODİ ALHUSSEIN and submitted on 08 /08/2022 has been **accepted unanimously** for the degree of Master of Science in Electrical and Computer Engineering

Prof. Dr. Galip CANSEVER

Supervisor

Thesis Defense Jury Members:

Prof. Dr. Galip CANSEVER	Faculty of Engineering and Architecture, Altinbas University	_____
Asst. Prof. Dr. Sefer KURNAZ	Faculty of Engineering and Architecture, Altinbas University	_____
Asst. Prof. Dr. Yüksel BAL	Faculty of Engineering Computer Engineering, Topkapi University	_____

I hereby declare that this thesis meets all format and submission requirements of a Master`s Thesis.

Submission date of the thesis to the Graduate Education Institute: ____/____/____

I hereby declare that all information presented in this graduation project has been obtained in full accordance with academic rules and ethical conduct. I also declare all unoriginal materials and conclusions have been cited in the text and all references mentioned in the Reference List have been cited in the text, and vice versa as required by the abovementioned rules and conduct.

Zinah Hayder Hammoodi ALHUSSEIN

Signature

DEDICATION

First, I would like to Allah Almighty for the power of mind, health, strength, guidance knowledge and skills to complete this study. This thesis is wholeheartedly dedicated to my beloved father, who have been my source of inspiration, he tells me that" every success in your life will be best gift for me". To my mother, who have been supporting me with the kind and pure love.

ACKNOWLEDGEMENTS

I would like to thank my supervisor Prof. Dr. Galip CANSEVER for all support during my study. It’s great pleasure to express my deepest gratitude to my friends who have shared with me best moments during my study for the master’s degree.

ABSTRACT

COMPUTER VISION AND DEEP LEARNING FOR AUTOMATIC DETECTION AND LOCALIZATION

Alhussein, Zinah

M.Sc., Electrical and Computer Engineering, Altınbaş University,

Supervisor: Prof. Dr. Galip CANSEVER

Date: 08/2022

Pages: 52

Machine learning techniques such as neural networks, which include concepts that have been found in organic nerve systems, are becoming more popular. A biological neuron is a cell that consists of a body, many smaller protrusions called dendrites, and a single long protrusion called an axon. Axons are the long protrusions that connect the dendrites to the body of the neuron. On the surface of the dendrites, there are several sites known as synapses, which are where axons from other neurons are joined to the neuron in question. An electric impulse is fired by the axons every now and then, and this changes the permeability of the cell membrane, causing a modest rise in the voltage within the neuron body. The greater the number of activations that come from other neurons, the greater the increase in voltage. A center is located at the base of the axon, and it is responsible for permanently measuring the voltage. The goal of semantic segmentation is to identify pixels belonging to the same object or class in the image and cluster them together. This method is valuable in multiple applications and projects such as detecting brain tumors, creating semantic maps with civilians for autonomous vehicles, face segmentation

or land usage classification. we used semantic segmentation, Semantic segmentation provides pixel-wise boundaries around screens, and as screens do not entirely overlap, it yields similar results as instance segmentation with significantly less complexity.

Keywords: Segmentation, ANN, Classification, Computer Vision

TABLE OF CONTENTS

Pages

LIST OF TABLES

Pages

LIST OF FIGURES

Pages

ABBREVIATIONS

ML	:	Machine Learning
NLP	:	Natural Language Processing
AI	:	Artificial Intelligence
NB	:	Naïve Bays
SVM	:	Support Victor Machine
LSI	:	Latin Semantic Index
GOM	:	Goals of Matrices

1 INTRODUCTION

The use of Artificial Neural Networks (ANN) in the area of neural network signal categorization may aid non-experts in interpreting the data and arriving at a diagnosis.

Artificial neural networks models, which are inspired by organic brain functioning, are utilized in machine learning. They are used to estimate functions that are dependent on a large number of unknown inputs. An artificial neural network (ANN) is a network of linked neurons that can communicate with one another. The associations have loads that are tweaked, making neural networks adaptable to the provided data and learning ready. The architecture of an ANN, which is basically the interconnection design between the layers of neurons, the learning procedure for refreshing loads of the interconnections, and the actuation work, which converts a neuron's weighted input information to its produced information enactment, are the three types of parameters that are displayed. Optimization is becoming the most important component of deep machine learning algorithms. It begins with the definition of a loss function/cost function and concludes with its minimization. A supervised or unsupervised learning methodology is used to teach or train an artificial neural network. By using supervised methods to update the weights on a regular basis, supervised learning trains the ANN. The supervised learning approach encompasses a wide range of algorithms.

1.1 THESIS CONTRIBUTION

We utilized the PyTorch [46] deep learning package and the fast.ai [28] framework for advanced learning cycles and U-Net architecture in the python programming language to create deep neural networks. In general, we used the OpenCV [32] and NumPy libraries to process photos. We used the Google Collaboratory [30] and Google Cloud research platforms, which offer both CPU and GPU computing capabilities. We employed semantic segmentation, which can be seen in Figure 1.1d, as one of the computer vision approaches shown in Figure 1.1. Semantic segmentation creates pixel-wise borders around screens, and since screens do not totally

overlap, it produces results that are comparable to instance segmentation but with much less complexity.

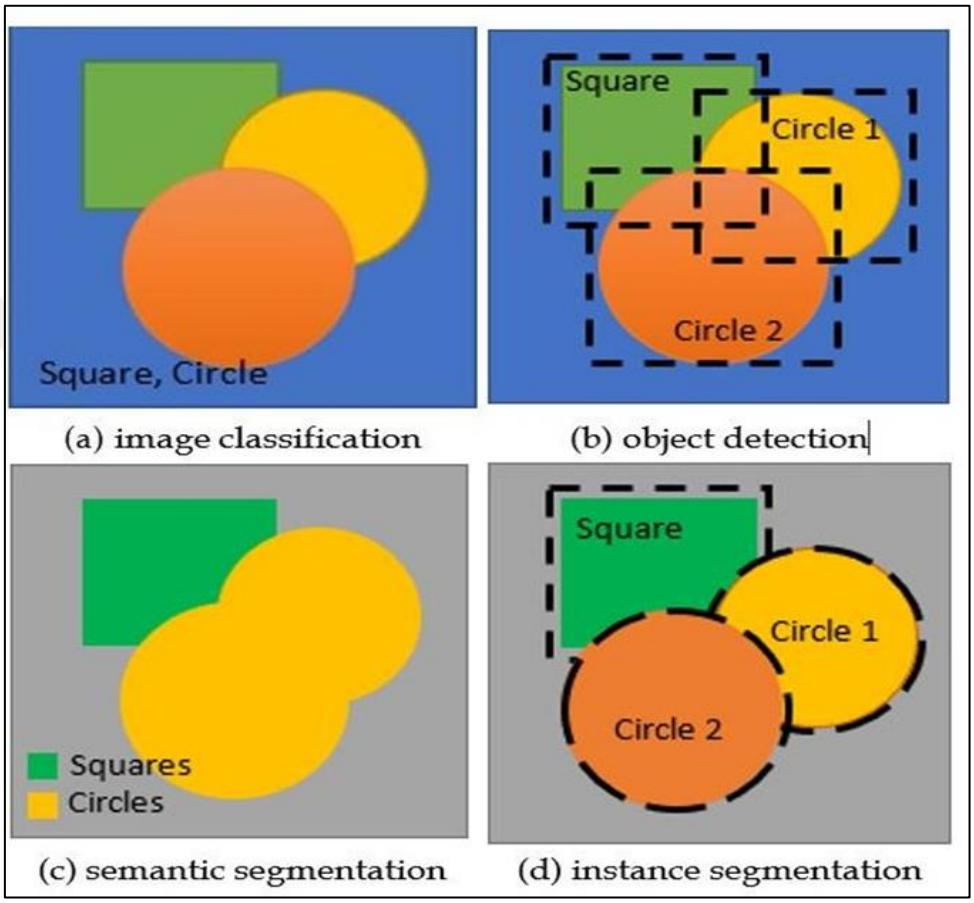

Figure 1.1: Computer vision problems categories.

2 NEURAL NETWORKS

2.1 INTRODUCTION

Artificial neural networks (ANNs) are inspired by the brain's organic neural connections. Artificial neurons (perceptrons) replace biological neurons in ANNs, which were initially described by Rosenblatt [53] in 1958. Multiple weighted inputs are added and transferred to an activation function like the sigmoid or threshold function in artificial neurons. The function of a neuron determines its output:

$$f(x) = \phi\ (x \cdot w) \tag{2.1}$$

where _x is the vector of features, w_ is the vector of weights, is a dot product operation and ϕ is the activation function. For a visual example, See Figure 2.2 for an illustration. The most frequent variant is to introduce bias, which is done by extending vector _x by one dimension and inserting 1 at the zero dimension. The vector w_ is likewise extended, so the bias is equal to the product of the initial elements in both vectors. The network consists of hierarchically ordered layers of neurons, The input and output layers are the first and final layers, respectively. There are hidden layers between the input and output layers. A hyperparameter is the number of hidden layers and the number of neurons in the hidden layers. The amount of features and predicted classes, respectively, determine the number of neurons in the input and output layers. The Multi-Layered Perceptron is a network that is made up of numerous layers of artificial neurons (perceptrons) (MLP).

Neurons from the previous layer send output to the inputs of neurons in the following layer via learnt weighted connections at prediction time. The feed-forward pass is what it's called.

A random weight initialization kicks off the learning process. The network then tries to optimize Gradient Descent (GD) or its variations with regard to a loss function in order to compute weight changes for each batch of data. Goodfellow et al. [23] offer a backpropaga- tion approach for computing weight updates.

Except for the design of ANNs, the learning rate is the most important hyper-parameter. What proportion of updates is used in the GD is determined by the learning rate. A network with a high

learning rate is more likely to vary about a minimum or even diverge, whereas a network with a low learning rate is more likely to converge slowly.

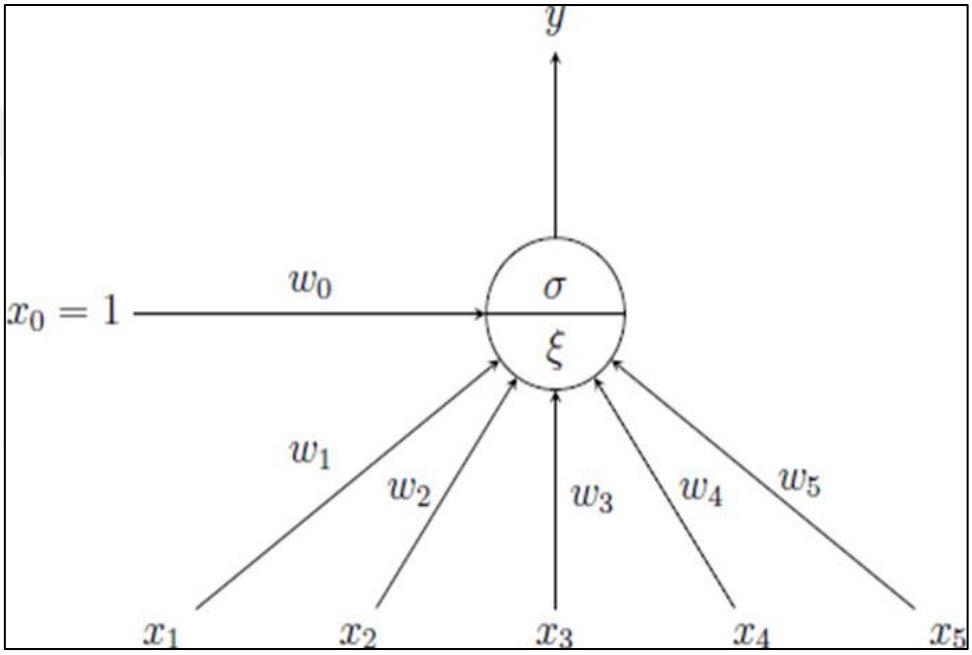

Figure 2.1: An example neuron unit with five inputs and bias.

2.2 PERCEPTRON

A perceptron, or simply a (artificial) neuron, is the fundamental component unit of most neural networks. It may be thought of as a basic representation of a organic neuron. A perceptron, In the same vein as its biological counterpart, is a form of 'black box' that receives a set number of inputs (sometimes referred to as dendrites) and produces just one output (also known as the axon). A related weight is assigned to each input. The weights reflect how much the connection has an influence on the final output. The following is a formalization of the above-mentioned

general notion. Let X = (x1.....xn) be the input value vector and w = (w1.....wn) be the weights vector. We must compute the weighted sum in order to calculate the outcome.

$$\xi = \mathbf{wX} = \sum^{n}_{i=1} wix \tag{2.2}$$

(Also known as the neuron's inner potential) is the first to be discovered. Following that, the weighted sum is transferred to the activation function _, which results in the output

$$y = \sigma(\xi) = \sigma\ (\sum^{n}{}^{i=1} \quad w_i x_i) \tag{2.3}$$

Depending on the purpose that the neuron is supposed to do, the activation function might be of many types. Threshold activation functions are commonly employed in (two-class) classification. A criterion. There are two sets of (two-dimensional) sample vectors, each of which is labeled 0 (red points) or 1 (blue points) on the x-axis (blue points). The current weight vector w(k) defines the h(k) hyperplane, which splits the vector space into two halves and defines the current weight vector w(k). samples in one half-space (the arrowheads on the graph) are categorized as 0, while samples in the other half-space (the arrowheads on the graph) are classified as 1.

$$\sigma(\xi) = \begin{cases} 1 & \xi \geq n \\ 0 & \xi < h \end{cases} \tag{2.4}$$

The beginning purpose has a generic form in where h is a real-valued parameter that can be arbitrary (but must be fixed). When the weighted total of inputs exceeds a pre-defined threshold value, the 'action potential' (i.e., non-zero output) is created, much like in a biological neuron.

A neuron with bias is a type of artificial neuron. An additional proper input x0, whose cost is continuously 1, is present in such a neuron. The weight w0 corresponds to this input. The product w0x0 = w0 is included in the weighted sum calculation, adding w0 to the weighted sum

of the other inputs is basically what this is. Suppose that w0 = h, where h is the previously set threshold, and that the initiation purpose is defined as

$$\sigma(\xi) = \begin{cases} 1 & \xi \geq 0 \\ 0 & \xi < 0 \end{cases} \tag{2.5}$$

This method is more practical for a variety of applications, including learning, as will be demonstrated in a moment. Unless otherwise stated, every usage of the term "neuron" is to be regarded as a neuron with bias.

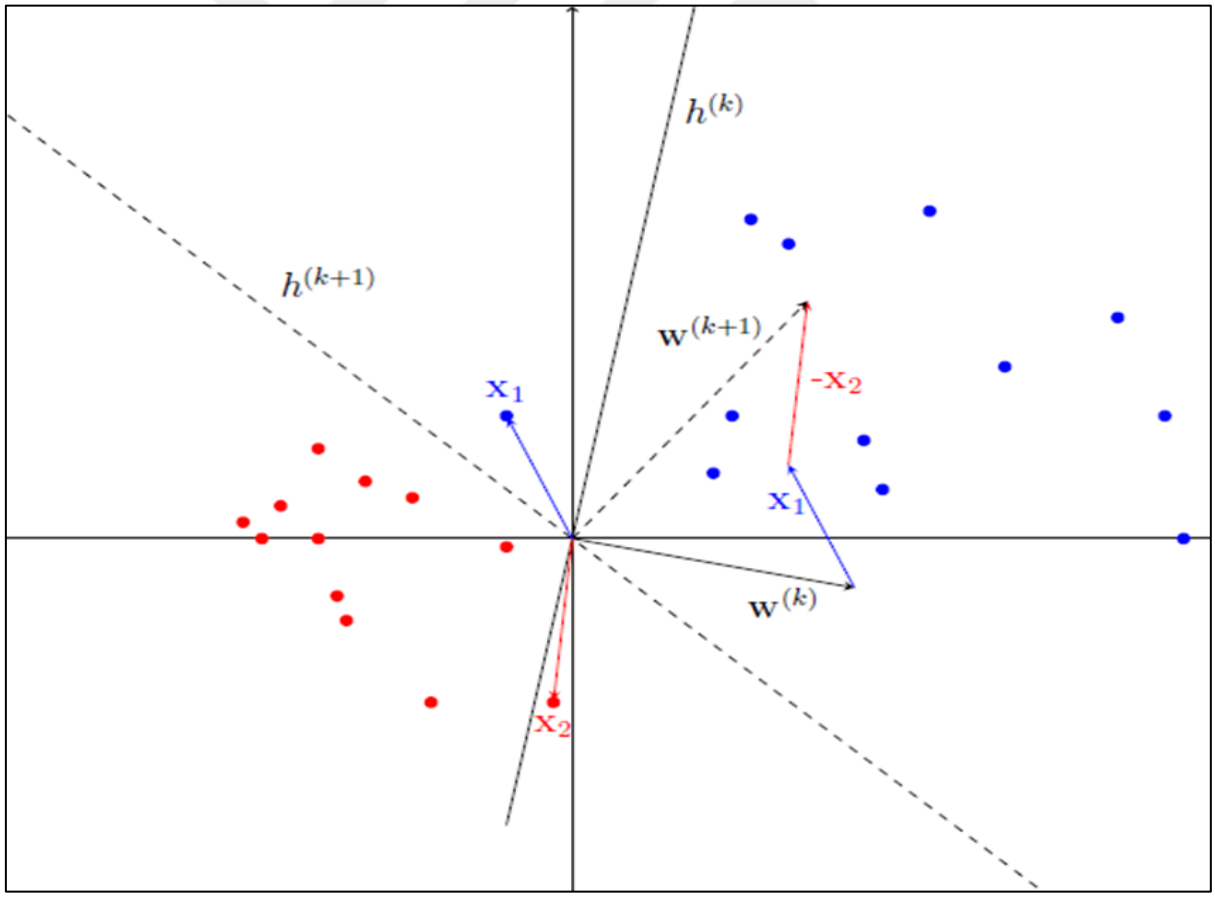

Figure 2.2: Geometric interpretation of perceptron classification and learning.

2.3 PERCEPTRON LEARNING ALGORITHM

A neuron may be seen as a naary purpose from Rn to the set f0; 1g, based on what has been described so far. It yields either 0 or 1 for each n-dimensional real vector, with the option chosen as stated in 3.2. The equation = w0x0 +. + wnxn = 0 is a (n - 1)- dimensional hyperplane as a result of which the space Rn is divided into two half spaces There are two half spaces: the first half space includes all vectors X for which £(X) = 0; and the second half space contains all vectors X for which £(X) = 0. The vector w = (w0.....wn) completely defines this hyperplane: the components (w1.....wn) describe the hyperplane's normal vector, while the words w0 w1.....w0 wn define its intersection points with the respective axes. The weights are increased in each succeeding stage.

$$\mathbf{w}^{(t)} = \mathbf{w}^{(t-1)} + \varepsilon \cdot \sum_{k=1}^{n} (d_k - \sigma(\mathbf{w}^{(t-1)}\mathbf{X}_k)) \cdot \mathbf{X}_k \tag{2.5}$$

If the sets of samples labeled with 0 and 1 are linearly separable, Frank Rosenblatt has previously proved that the approach always converges under these conditions. As a result, in the event that a hyperplane exists with all instances labeled as 0 on one side and all examples labeled as 1 on the other, the approach assures that it will be discovered in a reasonable period of time.

The criterion of linear separability places a substantial restriction on the types of data that may be used with the perceptron method. Not only does it fail to learn simple functions like the logical XOR, but it's also not robust enough to deal with outliers.

2.4 FAST PERCEPTRON TRAINING

The fundamental perceptron process, as stated in the preceding unit, creates a few issues that need be addressed first in instruction to improve the process's effectiveness.

To begin, what weights should be specified during the booting? Additional, in what way can the best value for the limitation ", i.e., learning rate, be determined? Is it preferable to keep it constant or adjust it with each iteration?

Gkanogiannis and Kalamboukis presented one solution to these problems (2009). The centers of both kinds of samples are computed first in their method, as follows:

$$\mathbf{C}_0 = \frac{1}{|C_0|} \sum_{\mathbf{x}_1 \in C_0} \mathbf{x_i} \tag{2.6}$$

$$\mathbf{C}_1 = \frac{1}{|C_1|} \sum_{\mathbf{x}_1 \in C_1} \mathbf{x_i} \tag{2.7}$$

The vector w = C1 - C0j is then used as the initial weight vector (w1…..wn). The method to find the initial weight vector is illustrated

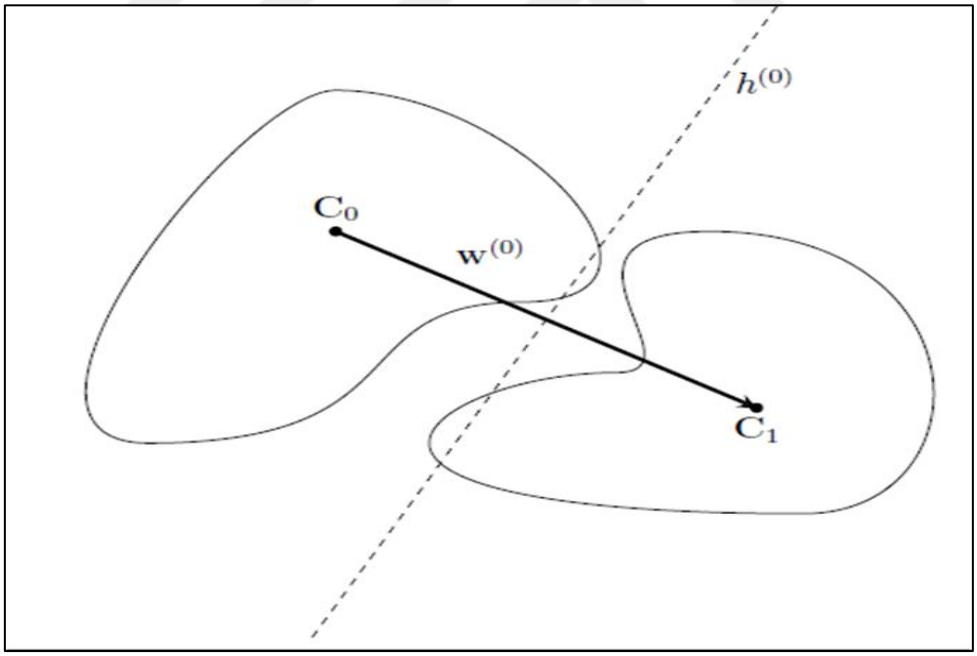

Figure 2.3: Initialization of weights.

For both classes of samples, the centers (C0; C1) were calculated. C1-C0 was then assigned as the starting weight vector w (0). Finally, the initial separation hyperplane h (0) was adjusted to

pass halfway between C0 and C1 by setting the bias w0 to C0+ C1-C02. Even if a few samples are still categorized wrongly in this configuration, a large number of samples have already been allocated to the proper class.

As seen in Figure 2.8. The authors recommend utilizing the Scut technique, which was first devised by [12], to determine the initial bias w0. To put it another way, it involves going through the training samples one by one, shifting the separating hyperplane to pass through the specific sample by selecting an acceptable bias value, and then c. calculates the 'quality' of such a setup (e.g., accuracy or F-measure). In the initialization, the best-scoring bias value is chosen as the real bias.

In addition to initialization, the authors propose a modified stepwise learning strategy for weight

adaption (see Figure 2.4).

In addition to initialization, the authors provide a modified stepwise learning strategy for weight adaption (see Figure 2.4).

The output values are computed for each sample in the standard technique, and the weights are then changed in such a way that the separating hyperplane is rotated in the direction of the misclassified samples. The parameter ", which must be determined experimentally, co-determines the amount to which the weights are adjusted. The modified variation is both like and unlike the original. It also uses the existing weights to determine the erroneously categorized samples of both types—false positives (FP) and false negatives (FN). Following that, the corresponding centers FP and FN are calculated as follows:

$$\mathbf{FP} = \frac{1}{|FP|}\sum_{x_k \in FP} x_k \tag{2.8}$$

$$\mathbf{FN} = \frac{1}{|FN|}\sum_{x_k \in FN} x_k \tag{2.9}$$

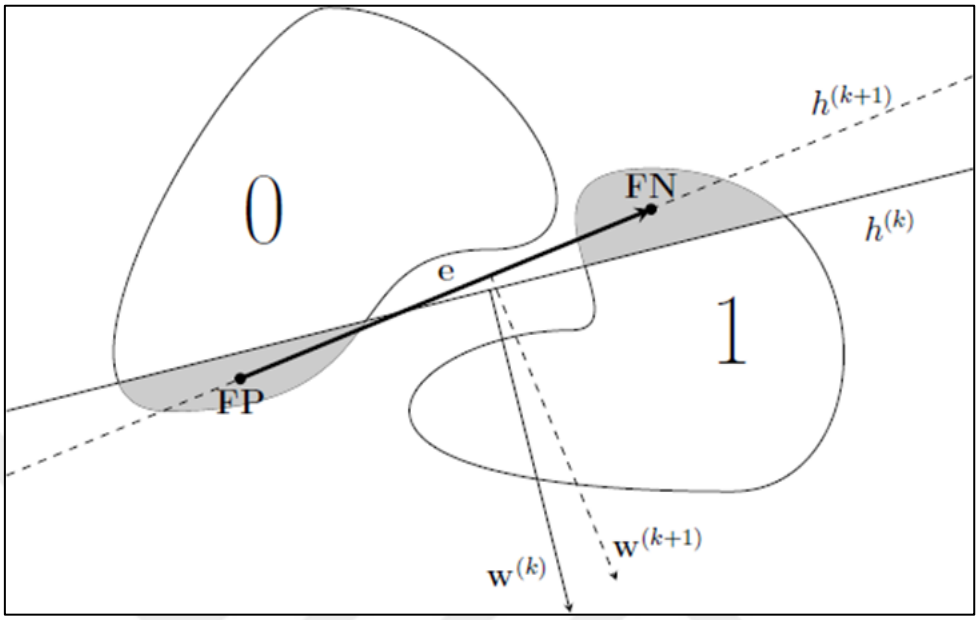

Figure 2.4: Illustration of one step in fast perceptron training.

We're looking for a hyperplane (or, in this case, a line) that divides two classes of cases. The separation line for the current weight vector w(k) is denoted by the letter h. (k). According to the data, some examples from session 0 are wrongly identified as 1 ("false positives" in the left older region), while others from class 1 are incorrectly classed as 0 ("false negatives" in the correct older area). For all false positives and false negatives, the centers FP and FN are determined, and then the error vector e is derived. Finally, the weights are adjusted so that FP and FN are both located on the new extrication hyperplane h(k+1).

The main advantage of the novel method is that the cost of " can be exactly computed. It is accomplished by assuming that a hyperplane traveling between the points FP and FN will result in a higher categorization rate. Because these sites are the centers of the misclassified sets, approximately half of the samples that were previously misclassified will now be accurately classified. (Of course, as a consequence of this, additional samples that were incorrectly classified may be discovered).

2.5 MULTI-LAYER FEED FORWARD NETWORKS

The production price was right used as the final outcome in the case of the single neuron unit described above, demonstrating the lesson given to the contribution vector. Instead of using the output, another approach is to utilize it as an input to another neuron. In this method, a network of linked neurons may be created, which is similar to biological neural networks in certain ways. For various objectives, a variety of topologies have been utilized in practice. The so-called multi-layer feedforward networks are one of the most extensively utilized types of neural networks. These are particularly important for this thesis since they have also been used in part-of-speech labeling. As a result, it's worth stating a few things about them.

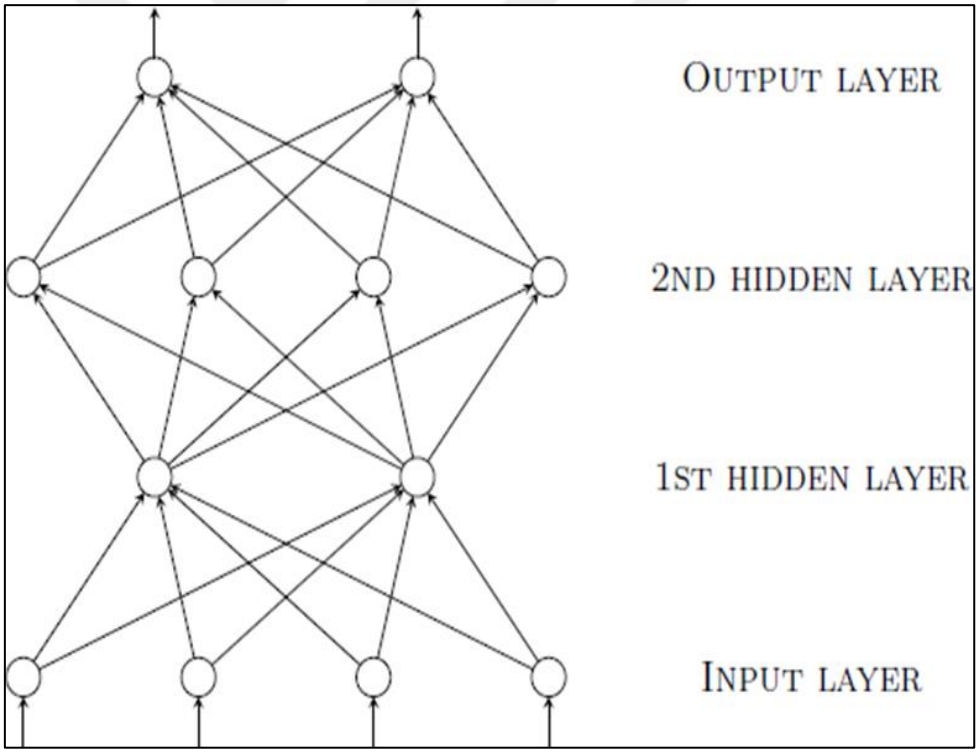

Figure 2.5: An example MLP network with two hidden layers.

(MLPs), are made up of a particular sum of nerve cell that are divided into numerous discontinuous groups, or layers, as the name suggests. Each MLP has a contribution layer that receives the contribution path and an production layer that generates a path of production values.

In addition, between the input and output levels, the network may have one or more hidden deposits. During the previous layer, the outputs of the neurons were used as inputs for the neurons in the subsequent layer, and the output is transmitted to the inputs of the neurons in the following layer.

Various activation functions can be employed, just as they can with a single perceptron unit. It must be differentiable for reasons that will be explained shortly. Because the threshold function (which is discontinuous in 0) cannot be utilized, other functions with a similar form are frequently used instead.

Similarly, to the threshold function, these functions work similarly in that they produce a value close to the maximum (1 in this example) for sufficiently large inputs; similarly, for sufficiently tiny inputs, they provide a value close to the minimum (0 in this example) (0, resp. -1). b. The functions rise continuously between these two extremes, with the growth rate determined by the parameter (in the logistic sigmoid) and the function a.

2.6 BACK-PROPAGATION LEARNING ALGORITHM

Study a multi-layer perceptron network with any number of layers. Let N be the total number of neurons in the system, IN N the number of neurons in the input layer, and OUT N the number of neurons in the output layer. Numbers ranging from 1 to jNj are used to index individual neurons. The weight of the connection between neurons I and j are represented by wji, W denotes the collection of all weights, whereas W denotes the collection of all weights. The notation I signifies the set of all neurons to which there is a connection from I, while the notation I denotes the set of all neurons to which i's input is coupled, respectively. After obtaining all of the output vectors yk, they must be compared to the desired output vectors dk to assess how well the network performs and how the weights should be adjusted to increase performance.

The squared error function increases in value when more training samples provide wrong outputs, and vice versa. The squared error is 0 if the network generates correct outputs for all

samples. The task of determining the appropriate network weights is thus identical to determining the global minimum of the function E.

Finding the global minimum by using the gradient E(w), which is equal to (wji)wjiW at a given location w and which corresponds to the current vector of all the weights wji in the network, is an approach. It is a vector that indicates the direction in which the error function develops the most quickly at a particular place, as well as the steepness of the slope in that direction. The gradient is defined as follows: Removing an existing weight vector's gradient produces a new vector that is likely to be "below" the original in terms of E value, as shown in Figure 1. This results in the vector w being updated on the tth iteration (t = 1; 2., etc.) of the loop.

3 BACKGROUND AND RELATED WORKS

3.1 DEEP LEARNING

The most significant moment and the beginning of Deep Learning using Convolutional Neural Networks was in 1995, when Yann LeCun and Yoshua Bengio took their steps from MLP and improved them by decreasing both the computing workload and the number of measurements. From that point forward, a publication was distributed [36] that demonstrates a more effective framework for acknowledgement by excluding the random variable. Then E. Hinton published a work in which he proposed Restricted Boltzmann Machines (RBMs) [37], which are used to filter, classify labeled and unlabeled data, and reduce data dimensionality, hence simplifying computational tasks. With the use of a convolutional neural network named Alex Net, Alex Krizhevsky and his colleagues were able to win the ImageNet LSVRC-2012, 2010 (Large Scale Visual Recognition Competition) using a train set of pictures in 2012.More than one million high-resolution RGB label photos were taken with the goal of categorizing them into 1000 different groups. The performance of DL algorithms has increased, and they may now be utilized to tackle a wide range of problems (image segmentation, detection, style transfer, analogies, and image classification). Because of the availability of massive volumes of data (known as "big data") and high-performance technology capable of performing large computing workloads, this progress has been made possible. Finally, in 2017, the researchers employed a deep neural network to conduct their research and discovered that the mistake rate was just 3 percent.

3.1.1 Deep Learning Overview

The chief meaning of DL is a Neural Network with several unseen sheets, therefore "profound" here refers to the number of hidden layers that exceeds two. It provides programmed highlight extraction by determining the qualities of information that may be used as a pointer to accurately mark information, with each layer removing highlights from the previous layer's output. This resembled a perplexing PC vision work, in contrast to a shallow system that needed a hand for

include extraction and extensive knowledge in the image preparation sector. However, switching from input photographs to vectors resulted in the loss of a large number of interesting data. In DL [38], there are seven main applications that have been refined to a high degree.

The performance of DL algorithms has increased, and they may now be utilized to tackle a wide range of problems (image segmentation, detection, style transfer, analogies, and image classification). This gain might be attributed to the availability of massive volumes of data (known as big data) as well as high-performance technology capable of solving difficult computational problems. Finally, in 2017, the researchers used a deep neural network, which had error rates of less than 3%.

3.1.2 Deep Learning Architectures Restricted Boltzmann Machines (RBMs)

An example of such a stochastic neural network is the hidden layer, which is comprised of two kinds of layers (visible layer and hidden layer), where every neuron in the visible layer is entirely connected with every neuron in the hidden layer, and vice versa for neurons in the hidden layer. Moreover, since there is no link between the neurons on the same layer, each neuron must make a stochastic choice about whether or not to transmit the signal across the two layers. As a result, connections are limited to connections where each input is a component of the connection.

RBMS is an unsupervised machine learning model that uses an energy base model to calculate the probability of a distribution given input data or to reduce energy consumption.

The input is sent from the Visible layer (input layer) to the Hidden layer through weights (w ij), and the product operation and activation function are performed on the Hidden layer, and update the weight by the equation:

$$\frac{\partial \log p(v^o)}{\partial w_{ij}} = (v i^o h j^o) - (v i^\infty h j^\infty) \tag{3.1}$$

Where i is visible unite, j is hidden unite, wij is the weight between the layers, ($v\ i^o h j^o$), ($v\ i^\infty h j^\infty$) are the correlations when (i,j) are in minimum and maximum layers.

The application of RBM is to lower the dimension, filter it, classify it, and extract features from it; however, the disadvantage is that it cannot acquire the partition function to the same resolution and is difficult to train.

Networks of Strong Beliefs (DBNs)

Each layer in DBN represents two levels of performance: a visible layer examined in relation to the one following it, and a hidden layer evaluated in relation to the layer before it. DBN Hinton offers a two-phase approach to resolve the unreliability of the network caused by the depth of the layers [41]: Phase 1: Layer-By-Layer

Taking the train When you enter data into an RBM and receive the parameters, the output will be utilized as input to the next RBM. This form will be a DBN that includes a large number of RBMs as well as the parameter that is used for feature extraction. Unsupervised learning is used to train the train in phase 1.

Phase 2: Finishing Touches

An easy-to-use classifier will be added at the conclusion of DBN by using the Contrastive Divergence CD method and feature extraction to construct an FFNN, which may be either BP or the DBN Softmax classifier, as appropriate.

The primary use of DBNs is in feature extraction, picture generation, grouping, and recognition, among other things.

Autoencoder (AE), also known as Auto associator, is an unsupervised learning and FFNN that consists of three main components (encoder, code, and decoder). After the data is shrunk in encoder, also known as latent-space representation, that present the code that transmit to decoder, which is rebuild the input data using this code transmit to code, the data is shrunk in encoder, also known as latent-space representation, that present the code that

Autoencoder (AE) is a tool that may be used to: a) reduce data in terms of dimensions.

b) Data compression is a technique for compressing data.

b) Put an end to data corruption.

d) By using a pre-train deep network, you may avoid the local minimum.

An example of a convolutional neural network (CNN or ConvNet) is shown below.

The following were the primary disadvantages of early neural networks:

When dealing with high-dimensional input, dealing with a large number of trainable parameters, even on a GPU, may be challenging (RGB).

Shifting, scaling, and other types of distortion are not sufficiently invariant or are not present at all. The topology of the input data should be ignored if there is no relationship between the pixels. c) When employing CNN, which is regarded a revolution in computer vision using mathematical operations, the following major unfavorable characteristics have been resolved: e) Invariance using scale, shift, and other similar operations; and Parameter sharing is used to solve overfitting problems by restricting the number of weights used in the model.

Create connections between neurons that are comparable to pixels that are close to each other, taking advantage of their spatial closeness to each other.

Because of parameter sharing and reduced connections, it classifies straight from image components with the least amount of treatment and the highest performance using a limited number of parameters. Because of automated feature extraction with little picture pre-processing, as well as parameter sharing and sparsity of connection, CNN has shown great accuracy and accomplishment in various sorts of study on image classification.

3.2 LONG SHORT-TERM MEMORY NETWORKS

Despite their widespread usage for sequential data, RNNs have certain drawbacks. RNNs, for example, are capable of remembering the past but not of keeping track of long-term dependencies. This issue arises as a result of the Vanishing Gradient and Exploding Gradient, which make RNN training challenging in a variety of ways. [31] goes into much information

about gradient difficulties. Long Short-term Memory Networks, on the other hand, solve this problem (LSTMs, [32]).

LSTMs are a special kind of RNN that can learn long-term dependencies. This benefit enables the network to retain vital information for extended periods of time while forgetting less critical information. This is why, in jobs like translation, voice recognition, and even music creation, this method is now quite effective. LSTM networks vary from standard RNNs in that they feature a repeating cell structure with four inter-acting layers.

The basic LSTM cell is shown in Figure 3.10. In actuality, there are several of them in different forms. The cell's most critical components are the cell state (represented by c) and the hidden state (represented by h). The input e represents the input at a certain timestamp t. While can be seen, the condition of the cell may remain unaltered as it passes through the cell.

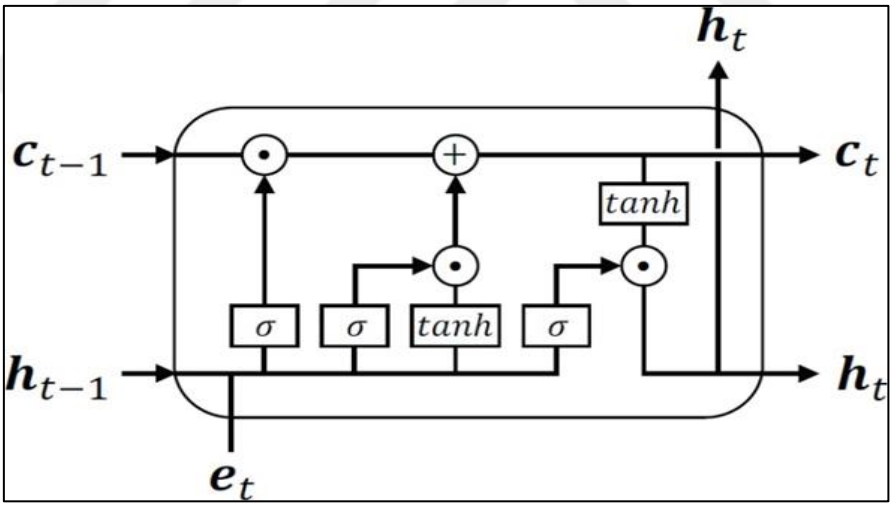

Figure 3.1: LSTM cell of the recurrent neural network [33].

Another important part of the cell are gates that manipulate with the cell memory. Gates are:

Forget gate – responsible for removing information from the cell state. It applies a sigmoid function σ for an input z and a hidden state h. Output of the sigmoid function is the vector with values from 0 to 1.

Input gate – resp· ansible for the addition of information to the cell state.

Output gate – responsible for what we are going to output from the cell. This decision is based on the current cell state.

LSTM networks also have some drawbacks. They take longer to train and require more memory during training. The downside is the requirement for many training data; otherwise, there is a possibility of overfitting. Those problems arise mainly because LSTM networks have more parameters comparing with other neural network architectures.

3.3 LITRATURE REVIEW

As demonstrated in Figure 2.1, the purpose of semantic segmentation is to identify pixels in an image that belong to the same item or class and cluster them together. This approach is useful in a variety of applications and research, including diagnosing brain tumors [13], building semantic maps for autonomous cars with citizens [57], face segmentation [34], and land use categorization [29].

A sliding window method is used to classify individual pixels. Each phase extracts characteristics from a window in the picture and classifies a subset of all pixels. As a consequence, a new picture of the same size is created, with classes represented by numbers that are mapped to a distinct color for better viewing. Annotating pixels with ground truth labels is the task's bottleneck. The kind of input data and the output provided by segmentation algorithms varies. Only supervised learning techniques with fixed classes and single picture input are considered in the next section.

To function successfully, semantic segmentation needs a large number of tagged pictures. Most datasets, on the other hand, are first provided with merely annotations for picture classification or object recognition and are only later enhanced for more sophisticated tasks with the cooperation of the community. The reason for this is because providing annotations for pixel-level tasks like semantic segmentation is more difficult.

The PASCAL Visual Object Classes (PASCAL VOC) [18] collection is well-known in the field of computer vision. It includes annotated photos with 21 semantic segmentation classes. This dataset serves as a baseline for most algorithms. Around 3,000 photos are separated into three sets: validation, training, and holdout test.

COCO [38], a more recent dataset with roughly 328k pictures with multi-purpose annotations centered on instance segmentation, is a more up-to-date dataset.

Cityscapes [8] is another recent dataset that focuses on urban environment interpretation for autonomous vehicle applications. The ImageNet Large Scale Visual Recognition Challenge (ILSVRC) image classification dataset has been updated for semantic segmentation, and the Cityscapes dataset has roughly 5k high-quality and 20k poor annotations divided into eight primary categories, as shown in Figure 2.1. To tackle these datasets, several state-of-the-art techniques have been presented in semantic segmentation contests.

Figure 3.2: Urban scene semantic segmentation from the CityScapes [8] dataset

3.4 TRADITIONAL TECHNIQUES

This section describes the traditional techniques for semantic segmen- tation that do not use deep learning. Traditional approaches involve feature extraction and classification as two independent steps. A more detailed summary of all traditional techniques for semantic segmenta- tion is given by Thoma [62].

3.4.1 Feature Extraction

The proper characteristics for the right application are critical to the algorithm's performance. There is no method for selecting the best candidates. Handcrafted features and specialist knowledge, such as edges or specialized patterns, are useful in certain applications.

The easiest method for calculating features is to use the pixel's color. They are compatible with grayscale (1 feature), RGB (3 features), HSI (3 features), and other color spaces. Because computer languages accommodate RGB features well, they are extensively utilized. HSI characteristics, on the other hand, might make the classifier more resistant to light. Examples of application may be found in Cohen et al. [7].

The Histogram of Oriented Gradients (HOG) [12] transforms the pixel position x, y into a color from the range 0,..., 255. A mapping function M (x, y) 0,...,255 may be used to express this. Following that, the original picture is split into two feature maps. The gradients of function M in x and y are represented by each map. The feature maps are then separated into patches, with a histogram calculated for each patch. For human detection, Bourdev et al. [5] employed HOG characteristics.

The SIFT (Scale-invariant Feature Transform) [39] describes global characteristics on an image around critical spot. Each key-point is surrounded by a 16-pixel patch that is split into 16 pixels-wide areas. HOG is calculated in each of these zones. A 128-dimensional vector emerges as a consequence of this process. Plath et al. [49] exploited SIFT features in their multi-class semantic segmentation on the PASCAL VOC [18] dataset. Histograms of a certain pattern count are histograms of the bag of visual words (BOV) [11] characteristics. Using the BOV visual

vocabulary, Csurka [9] converts SIFT features into high-level patch representation. On the PASCAL VOC 2007 dataset, our method outperformed the competition.

Edges and textons are examples of low-level picture characteristics. Textons are learnt in Convolutional Neural Networks' initial layers, according to Thoma [62]. (CNNs).

Classification Machine learning is responsible for the majority of the classifiers. We provide a list of the most often used feature classification classifiers in semantic segmentation:

Forest of Random Decisions [63]

A Random Decision Forest is a multi-decision tree ensemble method. A random subset of characteristics is used to train each tree.

(Support Vector Machine) (SVM)

A linear classifier is an SVM. The SVM may be expanded to greater dimensions using the kernel technique. The linear version attempts to maximize the difference between the prediction line and the parallel support vector crossing the nearest accurate classification.

MRFs (Markov Random Fields) [43]

MRFs are used to create a graphical model that is not directed. MRFs work on the principle of assigning a random variable to each feature and pixel. They are conditionally independent variables. All of these independencies are shown in the graph.

CRFs (Conditional Random Fields) [37]

MRFs with conditioned input characteristics are known as CRFs. In contrast to MRFs, CRFs do not make assumptions about input distribution and do not need input distribution computation.

Combining feature extraction and classification methods might result in a semantic segmentation model. Deep learning neural networks, on the other hand, have recently dominated most contests in terms of metrics.

3.4.2 Convolutional Neural Network

Fukushima [19], who called the CNN artificial neurons as neocog- nitrons, was the first to present them, and Krizhevsky et al. [36] popularized them with their AlexNet CNN design. CNNs are made up of convolutional, pooling, and ReLU layers that automatically extract important information and are usually followed by a fully connected layer for classification. VGG [58], Google Net [61], and ResNet [24] are some of the most often utilized CNN designs. Many deep learning algorithms for semantic segmentation are built on top of these frameworks.

3.4.2.1 Convolutional layer

Let the first layer's input be a 32-bit picture with three channels: red, green, and blue (see Figure 2.4a). Depending on stride (distance between sub-regions) and padding, we apply a single convolutional filter of size 3 x 3 x 3 by calculating the dot product with all feasible sub-regions (number of pixels placed around the image). As seen in Figure 2.4a, the outcome is a feature map.

Figure 2.4b shows the convolutional layer, which is made up of separate filters that generate a stack of feature maps. Parameter sharing refers to the sharing of weights across neurons in the same feature maps. The number of free parameters is reduced through parameter sharing, which takes use of the fact that the same feature might be helpful in several places.

Local connection is another crucial aspect in CNNs. In contrast to ANNs, where all neurons are completely linked, not all neurons are coupled to all inputs. Each neuron in a CNN links to a limited subset of input data; as a result, each neuron only sees a small portion of the picture rather than the whole image. This method keeps the image's spatial information.

3.4.2.2 Pooling layer

The pooling layer is responsible for non-linear down sampling. For example, max-pooling [51] takes the maximum value from each parti- tioned non-overlapping sub-region, see Figure 3.5.

3.4.2.3 ReLU layer

ReLU is the x max (0, x) ↦activation function, see Figure 2.6. ReLU preserves non-linear properties and its derivatives needed for GD are easily computed. It was first used by Glorot et al. [22].

3.4.3 ResNet

One of the most significant advancements in CNN design was ResNet [24]. Figure 2.7 shows how it creates residual blocks with skip connections. There are two pieces to the skip connection: The first section has two stacks of convolutional, ReLU, and pooling layers, whereas the second section bypasses these levels and adds the original input to the outcome of these layers. The goal is to keep inputs intact as they pass through the network2; as a result, a large number of layers may be stacked on top of each other. Figure 2.8 shows the comparison of VGG and ResNet.

3.4.4 Transfer Learning

As backbone feature extractors, many of the following algorithms leverage the aforementioned CNN pre-trained models. Models can distinguish various low-level patterns by training on strong datasets like ILSVRC. By retraining (also known as fine-tuning) a network, models with their gained information may be moved to a new domain. Because training big networks from scratch takes substantially longer than retraining them, the notion of retraining is widely employed. Advanced approaches include freezing and unfreezing layers in a network to enable just certain layers to be trained.

3.4.5 Fully Convolutional Network

Shelhamer et al. [56] advocated replacing fully connected classification layers with convolutional layers with 1 1 filters in previously trained classification models such as AlexNet [36] and ResNet [24]. These layers minimize the number of feature maps (dimensionality reduction) while still providing pixel-by-pixel forecasts. However, the projections were lower

than expected. The amount of shrinkage is determined on the stride and padding utilized. Figure 2.4b shows this impact with zero paddings and default stride.

The authors started by using linear up sampling to solve the issue. Another technique is to use skip layers: This method takes the last convoluted output, up samples it to the output size of the second last pooling layer, and pixel-wise averages them. The result, as well as the third and final pooling layer, are treated in the same way. Skip layers is based on the premise that up sampling may be taught. Figures 2.9 and 2.10 show that the results are more fine-grained than simple linear up sampling.

Shelhamer et al. [56] presented the Fully Convolutional Network (FCN) network, which obtained state-of-the-art results in PASCAL VOC and is considered a milestone in semantic segmentation. FCNs do, however, have several limitations: They don't take into account global background information, and inference takes a long time.

U-Net U-Net provides a u-shaped architecture in Figure 2.12 by combining FCN skip layers with encoder- decoder up sampling. Encoders and decoders convert data from one domain to another. As an encoder, most image recognition CNNs may be employed. The network extracts feature and builds a latent space representation in the encoder component. Figure 3.3 shows how the decoder conducts fractionally-stride convolution before concatenating output with the symmetric layer from the encoder. The size of the feature maps is increased via fractionally-stride convolution. Local pattern information is pre-served via feature maps transferred from the encoder to the decoder component.

The ISBI challenge 2015 for medical imaging was the first time U-Net was used. After then, U-Net was employed in fields other than medicine.

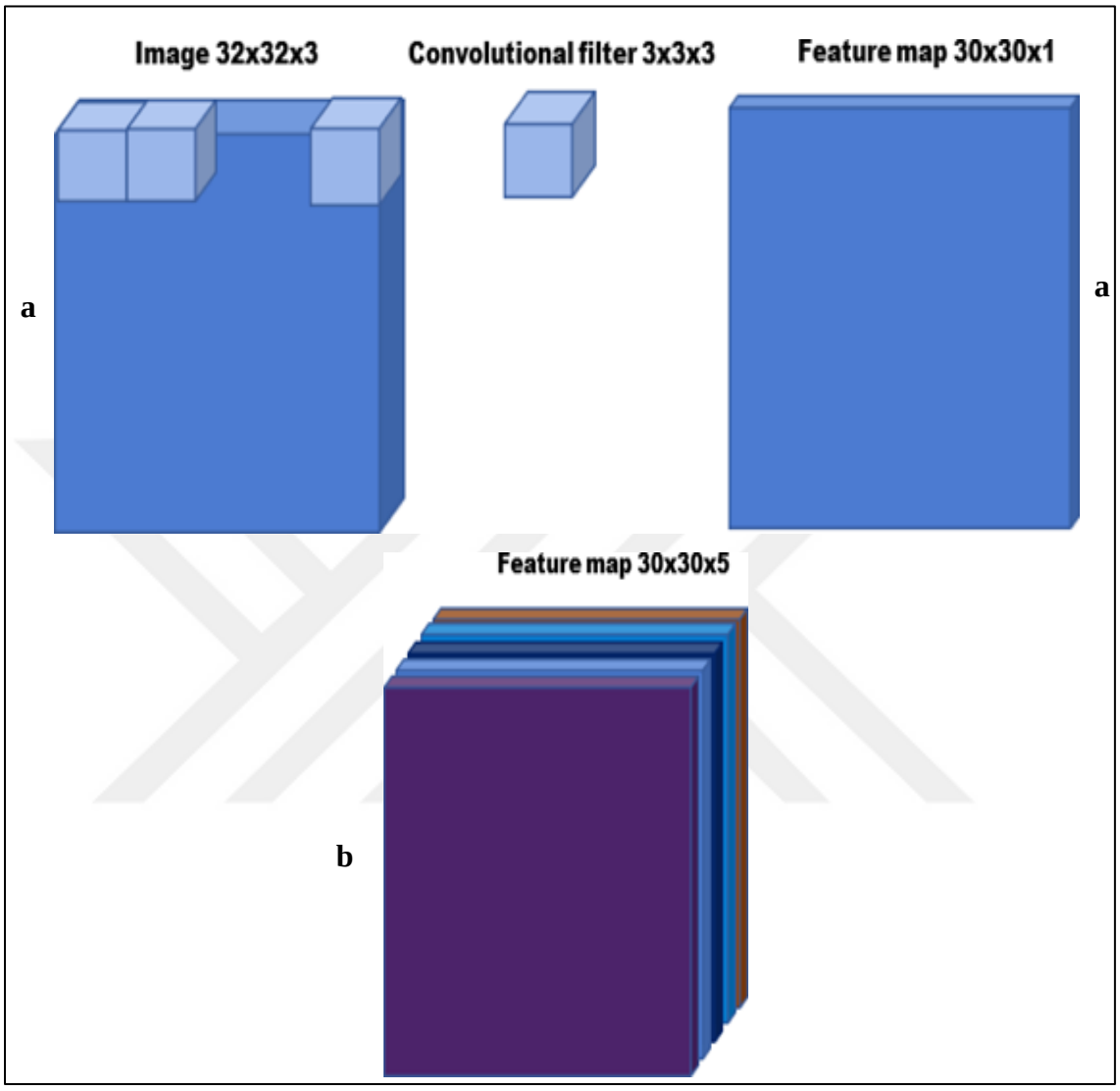

Figure 3.3: The effect of the convolutional layer on the size of the data. (a) When a single convolutional filter of size 3x3x3 (middle) is applied to the original image (left), the result is a feature map of size. (b) The result of five convolutional filters applied to the original im- age is and image of size 30 × 30 with five feature channels.

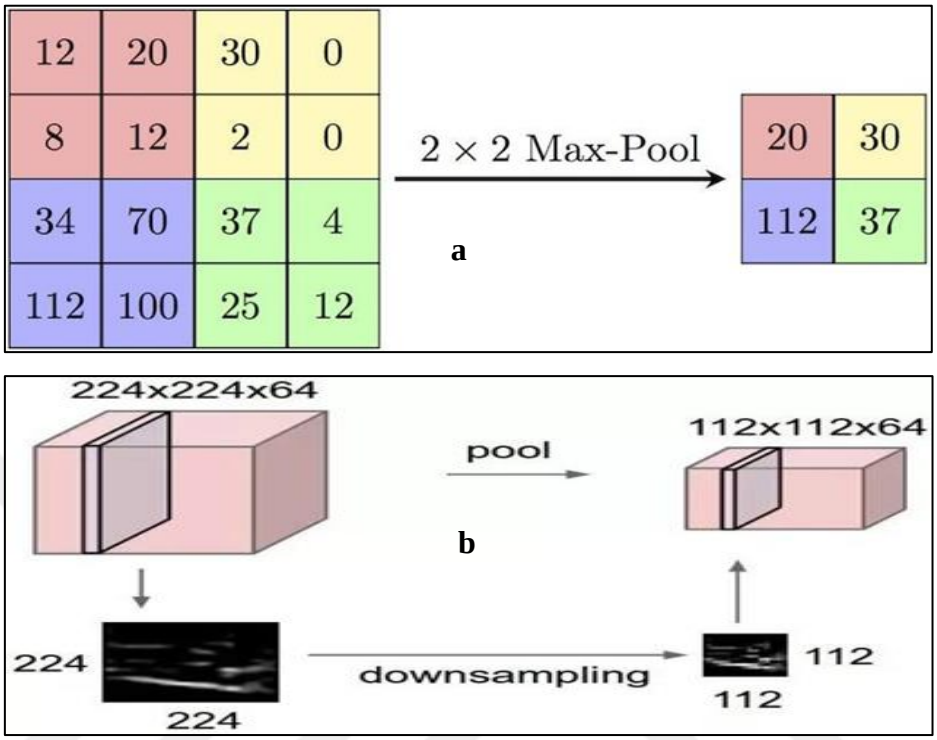

Figure 3.4: Max pooling (a) Max pooling from an original image with size 2*2 resulting in a down-sampled output (b) The effect of pooling on an image.

We wanted to find acceptable measures to indicate the quality of the models since our aim is to assess and compare models for object recognition. According to our findings, the most widely utilized statistic in several public contests is mean average precision (mAP) [17] [18] [19]. We will present an overview of the theoretical backdrop that we obtained from current literature study in this part. We'll go through how this measure is calculated and what it means.

We need to define several terminologies that come up while talking about object detection metrics. The first is the intersection of union and intersection (IOU). IOU is a score that indicates how accurate the anticipated location for a certain item is. It's calculated between two boxes that represent the image's objects. Typically, one box is the genuine location of the item that we want our models to infer, while the second box provides the model's actual prediction.

The area of intersection and union of the two boxes are used to calculate the score. Figure 2.1 depicts the junction and union area for the boxes. The score is then computed as:

$$IOU\ (\text{true box, predicted box}) = \frac{\text{area of intersection}}{\text{area of union}} \tag{3.2}$$

The IOU of 0 indicates that the two boxes do not overlap in any way. The IOU of 1 denotes the ideal circumstance, in which the two boxes perfectly overlap.

We consider the model's predictions on the provided picture when evaluating the object detector. A test dataset consisting of photos with manually annotated rectangular boxes defining the position and class of the box is required for assessment. The boxes in this test dataset will be referred to as the ground truth (GT).

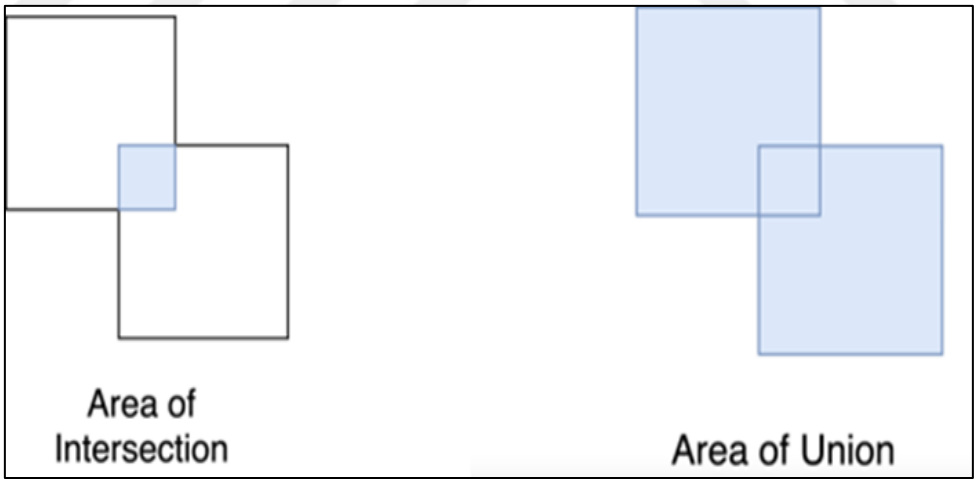

Figure 3.5: Illustration of the intersection and union area used for computing IOU

We want the object detector's predicted bounding boxes to be as near to the ground truth boxes as feasible. We distinguish between two circumstances when classifying predictions. It is considered a genuine positive if the projected box's IOU with the GT box exceeds the stated threshold (TP). If it's not true, it's called a false positive (FP). Furthermore, we consider false

negatives when the model fails to identify the ground truth box, which happens when none of the predicted boxes overlap with the GT box carrying the required IOU (FN). We don't usually include true negative (TN) in the object detection task, which implies not identifying a given bounding box, since there are so many options for box predictions that rewarding a model for not detecting them all isn't advantageous.

Accuracy and recall are critical for evaluating object detection models since a successful detector will have high precision and a large fraction of predicted boxes will be true positives. It will also have a strong recall, meaning it will recognize a large majority of the ground truth boxes. As a consequence, there will be very few false negatives. Simply expressed, accuracy refers to how effectively a model predicts just the most important variables (by having a low number of FP). The recall of the model tells us how successfully it identifies and detects all of the necessary boxes (by having a low number of FN). The accuracy and recall are calculated using equations (3.3).

$$\begin{aligned} \textbf{Precision} &= \frac{TP}{TP+FP} \\ \textbf{Recall} &= \frac{TP}{TP+FN} \end{aligned} \tag{3.3}$$

The precision/recall curve is a type of the plot where, on one axis, we have a recall, and on another axis, we have a precision. The curve tells us how the precision is changed when we increase recall. The precision of the good detector will stay high as we increase recall. However, it is fairly hard to tell which model performs better just by looking at the precision/recall curve of different models. Therefore, for evaluation, we usually use composite scores that try to summarize precision and recall, such as F1 score, average precision, or area under a curve (AUC).

In object detection competitions, it is common to use average pre- cision (AP) as the composite score. However, different competitions It is important to realize that average precision considers always bounding boxes just for one class. However, since object detectors are usually able to

detect more than one label, it is fairly common to use mean average precision (mAP). This score is computed as computing AP for each class that the object detector recognizes and then taking mean of all those APs. The resulting score summarizes precision and recall across all the classes.

4 METHODOLGY

4.1 METHODOLGY

This chapter provides a concise summary of the methodologies for data pre-processing, model formulation, and post-processing. The techniques of pre-processing are discussed in Section 4.2. In the next section (4.3), the model's specifics are broken down and discussed. Within the video699 [45] system, the post-processing techniques as well as the extraction of the four corner points that constitute the screen are discussed in Section 4.4. figure 4.1 shows the methodology activities.

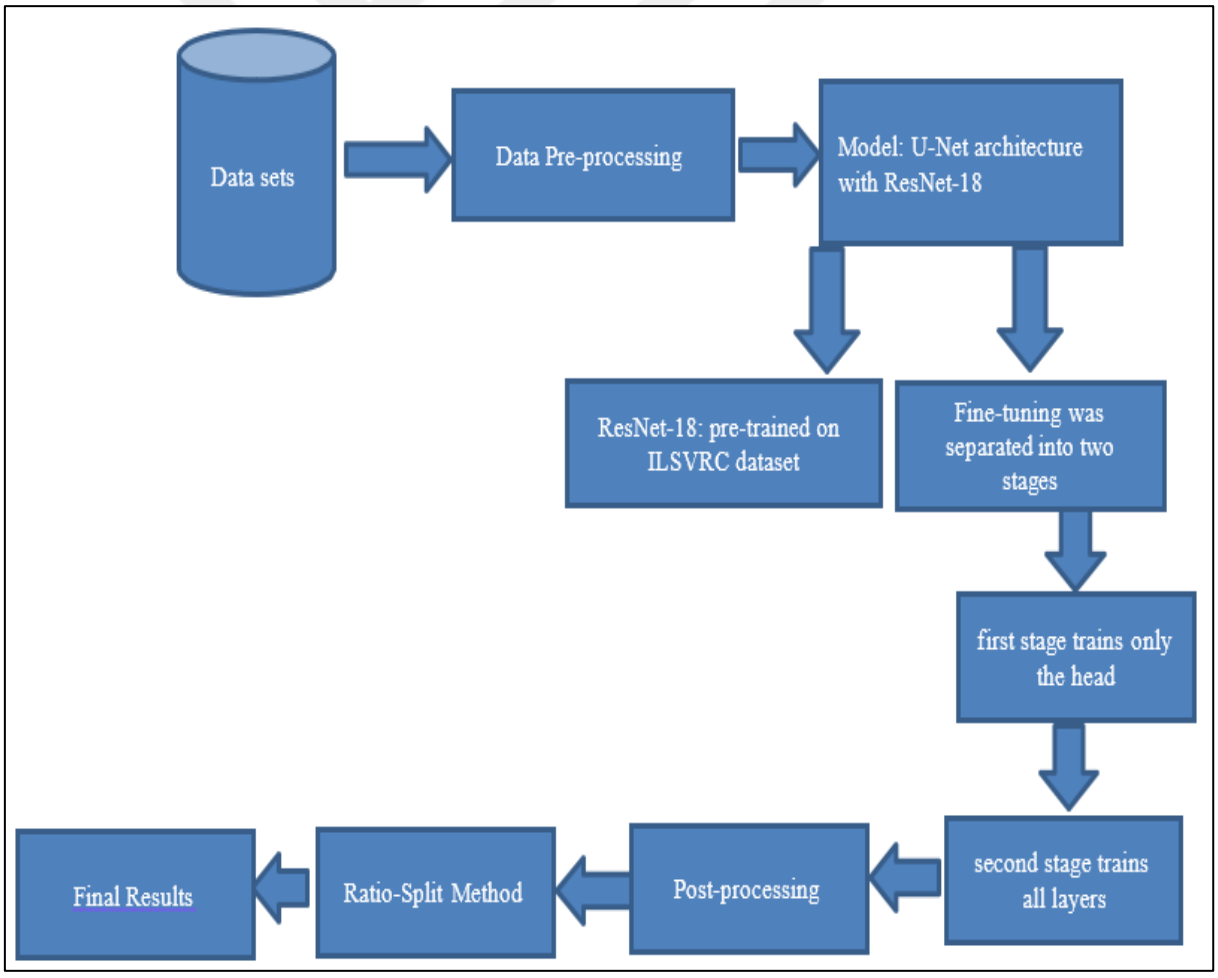

Figure 4.1: Research Methodology.

4.2 DATA PRE-PROCESSING

In order to make use of annotated coordinates as labels for semantic segmentation, a binary mask has to have those coordinates defined. As a result, we created binary masks by using the annotated

representations in Extensible Markup Language (XML). We used the fast.ai [28] platform in order to import data, parallelize calculations, and carry out changes in order to artificially extend the dataset and give improved generalization. The modification in brightness and contrast can be seen in Figure 3.2a, and the horizontal flipping transformation can be seen in Figure 3.2b. We used a probability of one for each of the transformations that we performed. As a result of the high memory requirements imposed by the design, the parameter re- size factor is used in order to linearly shrink pictures. We did not have the processing capabilities necessary to meet the minimum memory requirements for full photos; hence, we made the resize factor of 2 (that is, two times smaller) the default setting.

4.3 MODEL

We employed the architecture known as U-Net, with ResNet-18 serving as the backbone model. The ILSVRC dataset was used to perform preliminary training on ResNet-18. The process of fine-tuning was divided into two stages: the first stage trains just the head, which consists of the layers that are used for classification and up-sampling, and the second stage trains all of the layers.

4.4 POST-PROCESSING

In spite of the fact that the semantic segmentation process produces a binary picture as its output, the video699 system requires a screen detector to provide four corner coordinates for each and every illuminated projection screen that it finds. We came to the conclusion that the binary picture should have its linked components, also known as contours, extracted and then approximated using convex quadrangles.

We partitioned the binary picture into contours using the approach that was provided by Suzuki and Be [60], and we approximated the contours using the algorithm that was provided by Douglas and Peucker [15] and was implemented in the OpenCV [32] library. Both of these algorithms were employed.

In addition, post-processing is necessary in order to turn the polygons into convex quadrangles with four corner points, eliminate tiny contours, and divide overlapping contours. All of these steps must be performed. In the subsections that follow, distinct post-processing techniques, their impacts, and how sensitive they are to accurate parametrization will be discussed.

4.4.1 The Base Approach

It's possible that the output of semantic segmentation will include contours that are far too tiny to be considered screens. In the approach that we utilized as a starting point, we discarded contours that covered just a tiny fraction of the total pixels in a video frame by making use of the lower percentage border parameter. We also made use of the factor parameter in addition to the lower percentage limit. The greatest Hausdorff distance [1] from the original contour to the new polygon may be specified using the factor parameter, which, when multiplied by the perimeter of the contour, defines the shape of the new polygon. In our system, we made advantage of an iterable parameter to progressively loosen the condition up to the point when we retrieved either a quadrangle or a triangle, at which point we abandoned the triangle.

4.4.2 The Mechanism of Erosion-and-Dilation

In order to separate two illuminated projection screens that overlap and create a single shape, the morphological operators of erosion and dilation [47] are used. Erosion and dilation are two processes that are quite similar yet have opposite results. However, the input picture is altered when erosion and dilation operations are performed on it.

The erosion operator consists of moving a structuring element across the picture and replacing its center point with the minimum pixel value in the image pixels that are overlapped by the structuring element. This process is repeated until all of the image pixels have been affected.

The erosion operator is comparable to the dilation operator. On the other hand, it makes use of the pixel value that is the highest in the picture pixels that are overlapping those of the structuring element, the first step of the erosion-and-dilation procedure consisted of eroding the binary image that included the initial forecast in order to create shrunken projection screens. After that, we dilated each of the outlines that had been taken from the eroding forecasts one by one.

4.4.3 Method of Splitting by Ratio

Even using the erosion-and-dilation approach, it is possible that the illuminated projection screens cannot be successfully divided into different outlines when there is a large overlap between them. Our approach, which we call the ratio-split technique, consists of dividing the result obtained from the several earlier methods depending on the proportion of a contour's width to its height.

5 RESULTS

We predicted and achieved the best results from the image-wise splitting technique, as shown in Table 5.1, due to the inter-lecture variation. The lecture-wise splitting, on the other hand, comes the closest to a real-world application. Table 5.2 shows the resultant IoU, which is identical to an image- wise IoU but with a worse missing screens ratio. Unexpected screen positions and varied contexts in the test lectures might contribute to a higher rate of missing screens. To offer the model a glimpse into the new environment, we tried out the lecture peeping method. The missing screen ratio improved as a consequence, however IoU remained same. Because the lecture peeping method only needs the user to annotate one picture, it is much simpler than annotating the whole video and greatly increases screen detection quality.

We tested with the size of photos using the resize factor setting to get speed testing results. With a resize factor of 2, the GPU inference rate is 16 frames per second, whereas the CPU inference rate is just 2 frames per second. With a resize factor of 8, the CPU inference rate is 14 frames per second, whereas the GPU inference rate is 45 frames per second. Despite major improvements in CPU and GPU performance, and the CPU version now being capable of lower-quality films, the missed ratio rose, and IoU declined. A total of 7% of all screens were missed by the system.

Table 4.3 shows the optimized settings. According to the finding's, frozen epochs are more important to model performance than unfrozen epochs. Contours having an area of less than 6% of a picture were eliminated, and those with a width and height ratio of 0.3 were separated.

5.1 VALIDATION OF THE MODEL

The video699 Project dataset [44] comprises 17 lecture recordings that were selected at random from the lectures recorded at the Faculty of Informatics between 2010 and 2016. The authors selected a stratified sample of up to 25 video frames from each lecture recording for each lecture.

For training the model, choosing the post-processing techniques and their parameters, and measuring the model's performance, we employed nested 5-fold Cross-Validation (CV): The dataset is divided into the inner fold and the testing fold in the CV's outer loop. We separate the inner fold of the CV into the training fold and the validation fold in the inner loop. The training fold is used to train the model as well as to choose the post-processing techniques and parameters. To choose the optimal model, the validation fold is utilized. The testing fold is used to calculate the performance of the best model chosen by the CV's inner loop. The best model has the fewest missing frames and, as a result, the best IoU. The average performance over all five testing folds is reported. We employed three ways for partitioning the dataset into folds to explore the influence of inter-lecture variation on the chosen model and its performance:

a. The dataset is divided into lectures in the lecture-wise technique.

b. The dataset is divided into video frames in the frame-wise manner.

The dataset is partitioned into lectures in the lecture-wise peeking strategy, but the inner fold additionally comprises one video frame from each lecture in the testing fold and one video frame from each lecture in the validation fold.

Parameter optimization was used to choose parameters. Parameter optimization, unlike performance estimation, does not utilize nested CV and instead use the lecture-wise splitting method. The IoU and the number of missing frames are used to assess the model's performance, including post-processing.

Table 5.1: Performance estimation results with the image-wise splitting approach.

Metrics	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Mean
IoU	0.955	0.973	0.980	0.971	0.978	0.956
Missing	0.1	0.58	0.63	0.74	0.2	0.39

Table 5.2: Performance estimation results with the lecture-wise splitting approach.

Metrics	**Fold 1**	**Fold 2**	**Fold 3**	**Fold 4**	**Fold 5**	**Mean**
IoU	0.944	0.955	0.964	0.976	0.963	0.944
Missing	1.66	3.12	0.56	0.1	2.37	1.66

Table 5.3: Performance estimation results with the lecture-wise peeking splitting approach.

Metrics	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Mean
IoU	0.977	0.968	0.954	0.967	0.953	0.972
Missing	0.0	2.10	2.76	0.2	0.82	1.234

6 CONCLUSION

Neural networks are a kind of machine learning technology that uses concepts found in organic nerve systems. A biological neuron is a cell with a body, many smaller protrusions known as dendrites, and a single long protrusion known as an axon. Numerous places termed synapses may be discovered on the surface of the dendrites, where axons from other neurons are attached to the cell. The axons sometimes fire an electric impulse that changes the permeability of the cell membrane, causing the voltage within the neuron body to rise somewhat.

The purpose of semantic segmentation is to find pixels in an image that belong to the same item or class and group them together. This approach is useful in a variety of applications and research, including diagnosing brain tumors [13], building semantic maps for autonomous cars with citizens [57], face segmentation [34], and land use categorization [29].

We provide a short introduction of semantic segmentation techniques with an emphasis on deep learning methods in this paper. We used semantic segmentation to build and construct a lighted projection screen detecting system. We developed post-processing techniques to extract system-dependent structures from semantic segmentation results. We assessed the model using best practices, current assessment metrics, and industry standards. We demonstrated that even with a tiny dataset, screen detection is doable. With specific temporal limitations, the model may be employed in real-world applications: Our approach does not currently allow for real-time applications. We came to the conclusion that the ResNet backbone was maybe too sophisticated, and that a simpler model would be preferable.

REFERENCES

[1] Baltatzis, Vasileios, et al. "Bullying incidences identification within an immersive environment using HD EEG-based analysis: A Swarm Decomposition and Deep Learning approach." Scientific reports 7.1 (2017): 1-8.

[2] Abbas, Waseem, and Nadeem Ahmad Khan. "DeepMI: Deep learning for multiclass motor imagery classification." 2018 40th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC). IEEE, 2018.

[3] Tabar, Yousef Rezaei, and Ugur Halici. "A novel deep learning approach for classification of EEG motor imagery signals." Journal of neural engineering 14.1 (2016): 016003.

[4] Pereira, Arnaldo E., et al. "Cross-subject EEG event-related potential classification for brain-computer interfaces using residual networks." (2018).

[5] Vrbancic, Grega, and Vili Podgorelec. "Automatic classification of motor impairment neural disorders from EEG signals using deep convolutional neural networks." Elektronika ir Elektrotechnika 24.4 (2018): 3-7.

[6] Tang, Zhichuan, Chao Li, and Shouqian Sun. "Single-trial EEG classification of motor imagery using deep convolutional neural networks." Optik 130 (2017): 11-18.

[7] Dose, Hauke, et al. "An end-to-end deep learning approach to MI-EEG signal classification for BCIs." Expert Systems with Applications 114 (2018): 532-542.

[8] Wang, Zijian, et al. "Short time Fourier transformation and deep neural networks for motor imagery brain computer interface recognition." Concurrency and Computation: Practice and Experience 30.23 (2018): e4413.

[9] Dhanapal, R., and D. Bhanu. "Electroencephalogram classification using various artificial neural networks." Journal of Critical Reviews 7.4 (2020): 891-894.

[10] Antoniades, Andreas, et al. "Deep learning for epileptic intracranial EEG data." 2016 IEEE 26th International Workshop on Machine Learning for Signal Processing (MLSP). IEEE, 2016.

[11] Salama, Elham S., et al. "EEG-based emotion recognition using 3D convolutional neural networks." International Journal of Advanced Computer Science and Applications 9.8 (2018).

[12] Madden, Jamie M., et al. "Correlation between short-term blood pressure variability and left-ventricular mass index: a meta-analysis." Hypertension Research 39.3 (2016): 171-177.

[13] Qiao, Rui, et al. "A novel deep-learning based framework for multi-subject emotion recognition." 2017 4th International Conference on Information, Cybernetics and Computational Social Systems (ICCSS). IEEE, 2017.

[14] Goh, Sim Kuan, et al. "Spatio–spectral representation learning for electroencephalographic gait-pattern classification." IEEE Transactions on Neural Systems and Rehabilitation Engineering 26.9 (2018): 1858-1867.

[15] Kobler, Reinmar J., and Reinhold Scherer. "Restricted Boltzmann machines in sensory motor rhythm brain-computer interfacing: a study on inter-subject transfer and co-adaptation." 2016 IEEE international conference on systems, man, and cybernetics (SMC). IEEE, 2016.

[16] Zheng, Wei-Long, and Bao-Liang Lu. "Investigating critical frequency bands and channels for EEG-based emotion recognition with deep neural networks." IEEE Transactions on autonomous mental development 7.3 (2015): 162-175.

[17] Li, Feng, et al. "Deep models for engagement assessment with scarce label information." IEEE Transactions on Human-Machine Systems 47.4 (2016): 598-605.

[18] Huang, Jungming, Xiangmin Xu, and Tong Zhang. "Emotion classification using deep neural networks and emotional patches." 2017 IEEE International Conference on Bioinformatics and Biomedicine (BIBM). IEEE, 2017.

[19] Saleem, T. Mohamed, et al. "Aliskiren: A novel, orally active renin inhibitor." Systematic Reviews in Pharmacy 1.1 (2010): 93-93.

[20] Dhanapal, R., and D. Bhanu. "Electroencephalogram classification using various artificial neural networks." Journal of Critical Reviews 7.4 (2020): 891-894.

[21] Zhao, Yilu, and Lianghua He. "Deep learning in the EEG diagnosis of Alzheimer's disease." Asian conference on computer vision. Springer, Cham, 2014.

[22] Kulasingham, J. P., V. Vibujithan, and A. C. De Silva. "Deep belief networks and stacked autoencoders for the p300 guilty knowledge test." 2016 IEEE EMBS conference on biomedical engineering and sciences (IECBES). IEEE, 2016.

[23] Bablani, Annushree, Damodar Reddy Edla, and Venkatanareshbabu Kuppili. "Deceit identification test on EEG data using deep belief network." 2018 9th International Conference on Computing, Communication and Networking Technologies (ICCCNT). IEEE, 2018.

[24] Hajinoroozi, Mehdi, Zijing Mao, and Yufei Huang. "Prediction of driver's drowsy and alert states from EEG signals with deep learning." 2015 IEEE 6th international workshop on computational advances in multi-sensor adaptive processing (CAMSAP). IEEE, 2015.

[25] Li, Xiang, et al. "Emotion recognition from multi-channel EEG data through convolutional recurrent neural network." 2016 IEEE international conference on bioinformatics and biomedicine (BIBM). IEEE, 2016.

[26] Bashivan, Pouya, et al. "Learning representations from EEG with deep recurrent-convolutional neural networks." arXiv preprint arXiv:1511.06448 (2015).

[27] Bresch, Erik, Ulf Großekathöfer, and Gary Garcia-Molina. "Recurrent deep neural networks for real-time sleep stage classification from single channel EEG." Frontiers in computational neuroscience 12 (2018): 85.

[28] Dong, Hao, et al. "Mixed neural network approach for temporal sleep stage classification." IEEE Transactions on Neural Systems and Rehabilitation Engineering 26.2 (2017): 324-333.

[29] Hussein, Ramy, et al. "Optimized deep neural network architecture for robust detection of epileptic seizures using EEG signals." Clinical Neurophysiology 130.1 (2019): 25-37.

[30] Rawat, Vandana. "A classification system for diabetic patients with machine learning techniques." International Journal of Mathematical, Engineering and Management Sciences 4.3 (2019): 729.

[31] Alkhafaji, Ali Mohsin Lateef. Artificial neural networks for electroencephalogram classification. MS thesis. Altınbaş Üniversitesi/Lisansüstü Eğitim Enstitüsü, 2021.

[32] Zhang, Jianhua, and Sunan Li. "A deep learning scheme for mental workload classification based on restricted Boltzmann machines." Cognition, Technology & Work 19.4 (2017): 607-631.

[33] Dutta, Kusumika Krori, Poornima Sridharan, and Sunny Arokia Swamy Bellary. "Recurrent Neural Networks and Their Application in Seizure Classification." Deep Learning in Visual Computing and Signal Processing. Apple Academic Press, 2022. 165-203.

[34] Patnaik, Suprava, Lalita Moharkar, and Amogh Chaudhari. "Deep RNN learning for EEG based functional brain state inference." 2017 International Conference on Advances in Computing, Communication and Control (ICAC3). IEEE, 2017.

[35] Mammone, Nadia, et al. "Brain network analysis of compressive sensed high-density EEG signals in AD and MCI subjects." IEEE Transactions on Industrial Informatics 15.1 (2018): 527-536.

[36] Moretti, Davide V. "Theta and alpha EEG frequency interplay in subjects with mild cognitive impairment: evidence from EEG, MRI, and SPECT brain modifications." Frontiers in aging neuroscience 7 (2015): 31.

[37] Li, Gen, et al. "Deep learning for EEG data analytics: A survey." Concurrency and Computation: Practice and Experience 32.18 (2020): e5199.

[38] Ieracitano, Cosimo, et al. "A time-frequency based machine learning system for brain states classification via eeg signal processing." 2019 International Joint Conference on Neural Networks (IJCNN). IEEE, 2019.

[39] Fiscon, Giulia, et al. "Alzheimer's disease patients classification through EEG signals processing." 2014 IEEE Symposium on Computational Intelligence and Data Mining (CIDM). IEEE, 2014.

[40] Tsiouris, Kostas M., et al. "A long short-term memory deep learning network for the prediction of epileptic seizures using EEG signals." Computers in biology and medicine 99 (2018): 24-37.

[41] Nguyen, Minh, et al. "Modeling Alzheimer's disease progression using deep recurrent neural networks." 2018 International Workshop on Pattern Recognition in Neuroimaging (PRNI). IEEE, 2018.

[42] Liu, Lin, et al. "A new machine learning method for identifying Alzheimer's disease." Simulation Modelling Practice and Theory 99 (2020): 102023.

[43] Chowdhury, Tanima Tasmin, et al. "Seizure and non-seizure EEG signals detection using 1-D convolutional neural network architecture of deep learning algorithm." 2019 1st International Conference on Advances in Science, Engineering and Robotics Technology (ICASERT). IEEE, 2019.

[44] Al-kaysi, Alaa M., Ahmed Al-Ani, and Tjeerd W. Boonstra. "A multichannel deep belief network for the classification of EEG data." International conference on neural information processing. Springer, Cham, 2015.

[45] Kim, Donghyeon, and Kiseon Kim. "Detection of early stage Alzheimer's disease using EEG relative power with deep neural network." 2018 40th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC). IEEE, 2018.

[46] Moinnereau, Marc-Antoine, et al. "EEG artifact removal for improved automated lane change detection while driving." 2018 IEEE International Conference on Systems, Man, and Cybernetics (SMC). IEEE, 2018.

[47] Qi, Shuhao, et al. "Recognition of composite motions based on sEMG via deep learning." 2019 14th IEEE Conference on Industrial Electronics and Applications (ICIEA). IEEE, 2019.

[48] Sandheep, P., et al. "Performance analysis of deep learning CNN in classification of depression EEG signals." TENCON 2019-2019 IEEE Region 10 Conference (TENCON). IEEE, 2019.

[49] Yıldırım, Özal, Ulas Baran Baloglu, and U. Rajendra Acharya. "A deep convolutional neural network model for automated identification of abnormal EEG signals." Neural Computing and Applications 32.20 (2020): 15857-15868.

[50] Shao, Hui-Min, et al. "EEG-based emotion recognition with deep convolution neural network." 2019 IEEE 8th Data Driven Control and Learning Systems Conference (DDCLS). IEEE, 2019.

[51] Lu, Na, et al. "A deep learning scheme for motor imagery classification based on restricted Boltzmann machines." IEEE transactions on neural systems and rehabilitation engineering 25.6 (2016): 566-576.

[52] Alramdani, Adari Taher Salem. Brain tumor segmentation and classification using machine learning techniques. MS thesis. Altınbaş Üniversitesi/Lisansüstü Eğitim Enstitüsü, 2022.

[53] Babu, G. Stalin. "Exploiting of Classification Paradigms for Early diagnosis of Alzheimer's disease." INFORMATION TECHNOLOGY IN INDUSTRY 9.2 (2021): 281-288.

[54] Zuo, Haiqiang, et al. "Combining convolutional and recurrent neural networks for human skin detection." IEEE Signal Processing Letters 24.3 (2017): 289-293.

[55] Tabarestani, Solale, et al. "Longitudinal prediction modeling of alzheimer disease using recurrent neural networks." 2019 IEEE EMBS International Conference on Biomedical & Health Informatics (BHI). IEEE, 2019.

[56] Forouzannezhad, Parisa, et al. "A deep neural network approach for early diagnosis of mild cognitive impairment using multiple features." 2018 17th IEEE international conference on machine learning and applications (ICMLA). IEEE, 2018.

[57] Orimaye, Sylvester Olubolu, Jojo Sze-Meng Wong, and Judyanne Sharmini Gilbert Fernandez. "Deep-deep neural network language models for predicting mild cognitive impairment." BAI@ IJCAI. 2016.

[58] Ullah, Ihsan, Muhammad Hussain, and Hatim Aboalsamh. "An automated system for epilepsy detection using EEG brain signals based on deep learning approach." Expert Systems with Applications 107 (2018): 61-71.

[59] Acharya, U. Rajendra, et al. "Deep convolutional neural network for the automated detection and diagnosis of seizure using EEG signals." Computers in biology and medicine 100 (2018): 270-278.

[60] Yuan, Han, and Bin He. "Brain–computer interfaces using sensorimotor rhythms: current state and future perspectives." IEEE Transactions on Biomedical Engineering 61.5 (2014): 1425-1435.

[61] Zhang, Daoqiang, Dinggang Shen, and Alzheimer's Disease Neuroimaging Initiative. "Multi-modal multi-task learning for joint prediction of multiple regression and classification variables in Alzheimer's disease." NeuroImage 59.2 (2012): 895-907.

[62] Moradi, Elaheh, et al. "Machine learning framework for early MRI-based Alzheimer's conversion prediction in MCI subjects." Neuroimage 104 (2015): 398-412.

[63] Zhang, Xiuming, et al. "Bayesian model reveals latent atrophy factors with dissociable cognitive trajectories in Alzheimer’s disease." Proceedings of the National Academy of Sciences 113.42 (2016): E6535-E6544.