PEOPLE'S DEMOCRATIC REPUBLIC OF ALGERIA

MINISTRY OF HIGHER EDUCATION AND SCIENTIFIC RESEARCH

ECHAHID HAMMA LAKHDAR UNIVERSITY EL OUED

EXACT SCIENCES FACULTY

COMPUTER SCIENCE DEPARTMENT

THESIS

In Candidacy for the Degree of

DOCTOR 3rd CYCLE LMD IN COMPUTER SCIENCE

By: Meriem DJOUADI

Improving location recognition based on multiple images transmitted from a user's phone to aid in way-finding

---

Defended on December 17, 2024.

Examination Committee:

Chairman: Pr. Mohammed Charaf Eddine MEFTAH, Professor – El Oued University

Supervisor: Pr. Mohamed-Khireddine KHOLLADI, Professor – El Oued University

Examinator: Dr. Oussama AIADI, MCA – Kasdi Merbah University of Ouargla

Examinator: Dr. Mourad Bouzenada, MCA – Université Abdelhamid Mehri Constantine2

Examinator: Dr. Soltane MERZOUG, MCA – Larbi Tebessi University of Tébessa

Examinator: Pr. Abderkader LAOUID, Professor – El Oued University

El Oued, Algeria, 2024

DEDICATION

~

To my parents, sisters and brother

To my husband and children

~

ACKNOWLEDGEMENT

First and foremost, I am very grateful to Allah for granting me the opportunity and the ability to proceed successfully.

I would like to express my sincere gratitude to my supervisor, Prof. Mohamed-Khireddine KHOLLADI, for his invaluable advice, continuous support, and patience during my PhD study. His immense knowledge and plentiful experience have encouraged me in all the time of my academic research and daily life.

I would like to thank the jury members for honoring me with their judgment of my work, and I appreciate their valuable comments, thoughtful discussions, and intellectual input into this research.

I would like also to thank my friends, lab mates, colleagues and research team.

Finally, I am grateful to my family - my parents, husband, sisters, and brother. Their unwavering support and understanding have been integral in helping me complete my studies over the past few years.

ABSTRACT

Image geo-localization involves determining the geographical location where a particular image was taken using only its visual information, without relying on any associated metadata. This process often requires advanced techniques in computer vision and machine learning also analyzing features within the image to establish its precise geographical coordinates. Predicting images locations is one of the most challenging and difficult tasks with numerous real-world applications. Lately, significant research has been dedicated to recognizing places and geo-locating images to estimate geographic locations and identify user positions. Recently, Convolutional Neural Networks have shown their capacity to extract significant features in various domains, particularly in computer vision where Deep Learning surpassed previous state-of-the-art methods.

Our principal contribution in this dissertation is assessing the use of the extracted features from Pre-trained CNN models and Support Vector Machine (SVM) to improve Street View image classification performance. We assessed our proposed methods on the dataset of Google Street View (GSV), which consists of high-resolution images from the city center as well as the neighboring regions of Pittsburgh, Orlando, and partially Manhattan. The experiment results proved that our proposed approaches significantly improve the accuracy of classification.

Keywords: location recognition, image geo- localization, classification, pre- trained CNN models, Google Street View images (GSV), Support Vector Machine (SVM).

RÉSUMÉ

La géolocalisation d'images consiste à déterminer l'emplacement géographique où une image particulière a été prise en utilisant uniquement ses informations visuelles, sans s'appuyer sur les métadonnées associées. Ce processus nécessite souvent des techniques avancées de vision par ordinateur et d'apprentissage automatique analysant également les caractéristiques de l'image pour établir ses coordonnées géographiques précises. La prédiction de l'emplacement des images est l'une des tâches les plus difficiles avec de nombreuses applications réelles. Dernièrement, d'importantes recherches ont été consacrées à la reconnaissance de lieux et à la géolocalisation d'images pour estimer les emplacements géographiques et identifier les positions des utilisateurs. Nouvellement, les réseaux de neurones convolutifs ont montré ses capacités à extraire des caractéristiques significatives dans divers domaines, en particulier en vision par ordinateur où l'apprentissage en profondeur a dépassé les méthodes de pointe précédentes.

Notre principale contribution dans cette thèse est d'évaluer l'utilisation de caractéristiques extraites de modèles CNN pré-entraînés et de Machines à vecteurs de support (SVM) pour améliorer les performances de classification des images Street View. Nous avons évalué nos approches proposées sur l'ensemble de données Google Street View (GSV), qui contient des images de haute qualité du centre-ville ainsi que des zones voisines de Pittsburgh, Orlando et partiellement Manhattan. Les résultats des expériences ont prouvé que notre méthode proposée améliorer significativement l'exactitude de la classification.

Mots clés: reconnaissance de localisation, géolocalisation d'images, classification, modèles CNN pré-entraînés, images Google Street View (GSV), Machine vectorielle de support (SVM).

الملخص

يتضمن تحديد الموقع الجغرافي للصور تحديد الموقع الجغرافي الذي التقطت فيه صورة معينة باستخدام معلوماتها المرئية فقط، دون الاعتماد على أي بيانات وصفية مرتبطة بها. تتطلب هذه العملية في كثير من الأحيان تقنيات متقدمة في الرؤية الحاسوبية والتعلم الآلي أيضًا تحليل الميزات داخل الصورة لتحديد إحداثياتها الجغرافية بدقة. يُعد التنبؤ بموقع الصور إحدى أكثر المهام صعوبة مع العديد من التطبيقات في العالم الحقيقي. في الآونة الأخيرة، خُصّصت بحوث مهمة للتعرف على الأماكن وتحديد المواقع الجغرافية للصور من أجل تقدير المواقع الجغرافية وتحديد مواقع المستخدمين. في الآونة الأخيرة، أظهرت الشبكات العصبية التلافيفية قدرتها على استخراج ميزات مهمة في مجالات مختلفة، لا سيما في مجال الرؤية الحاسوبية حيث تفوقت أساليب التعلم العميق على أحدث الأساليب السابقة.

تتمثل مساهمتنا الرئيسية في هذه الأطروحة في تقييم استخدام الميزات المستخرجة من نماذج CNN المدربة مسبقًا وآلة دعم المتجهات (SVM) لتحسين أداء تصنيف صور التجوّل الافتراضي. قمنا بتقييم نهجنا المقترح على مجموعة بيانات Google Street View (GSV)، والتي تحتوي على صور عالية الدقة من وسط المدينة بالإضافة إلى المناطق المجاورة في بيتسبرغ وأورلاندو وجزء من مانهاتن. أظهرت نتائج التجارب أن طريقتنا المقترحة يمكنها تحسين دقة التصنيف بشكل كبير.

الكلمات المفتاحية: التعرف على الموقع، تحديد الموقع الجغرافي للصور، التصنيف، نماذج CNN المدربة مسبقًا، صور Google Street View (GSV) ، آلة دعم المتجهات (SVM).

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

GENERAL INTRODUCTION

GENERAL INTRODUCTION

The increasing prevalence of smartphones and digital devices has led to a rising need for innovative applications capable of leveraging their built-in cameras and Global Positioning System (GPS). This includes personal photography, enhanced reality experiences and navigation based on vision. The increasing volume of visual and digital data generated through smartphone cameras or other devices, as well as publicly available datasets from different platforms, indicates potential for innovative image-based applications. Furthermore, the use of smartphone cameras also opens up possibilities for providing datasets without GPS coordinates. As more people use digital devices and their capabilities, there is a growing need for new approaches to understand and interpret the vast amount of visual information related to places depicted in images captured through these devices. Moreover, these places could range from landmarks and tourist destinations to cities and natural landscapes.

Image geo-localization, also known as location estimation, location inference, geo-tagging, geo-locating, and geospatial localization, involves the process of inferring the actual geographic locations at which images were taken. This mission usually includes analyzing and extracting image content or associated metadata to identify geographical information. Despite various methodologies for image geo-localization, it remains a largely unsolved challenge in the field, where there are two main approaches to handling it; the problem is often referred to a challenge of image retrieval. Within that context, the existing knowing of the specific locations relevant to the function is depicted through a set of images. Every image in this set (database) has an associated identifier indicating its geographical location, such as the GPS coordinates or the name of a prominent landmark. When there is a need to determine the location of a new picture (query), the framework of location recognition extensively searches the database for visually similar images. Upon finding identical images, their tagged locations are employed to deduce the location query. Otherwise, the problem is formulated as a classification where features are extracted from the input images and then used to train a classifier that predicts the image locations. The efficiency of classification-based approaches relies heavily on the feature extraction techniques capabilitie.

Predicting the geographic location and estimating where the image was taken based only on its pixel content (visual data) represents a highly challenging task. However, by

leveraging machine learning techniques and deep learning algorithms, the academic community has achieved considerable progress in solving this problem. Recent advancements in machine learning and computer vision have enabled the research community to develop novel algorithms for analyzing and understanding images with greater precision and efficiency. These algorithms can automatically detect and classify objects, landmarks, and features within the images, providing valuable location recognition and accurate geographical information.

Lately, Convolutional Neural Networks (CNNs) have demonstrated their ability to obtain powerful features that play a crucial role in improving performance. As a result, the effectiveness of this approach allows image geo- localization significantly advantage from the feature extraction capabilities of CNNs, significantly improving the classification accuracy. Through the utilization of the functional capabilities of CNNs, image geo-localization can overcome its unsolved nature and achieve higher accuracy in forecasting the geographic location of images.

Our primary focus in this dissertation is to evaluate the utilization of characteristics obtained from various pre- trained CNN models and fed them into the classifier of Support Vector Machine can ameliorate the performance of Street View image classification. This approach leverages deep feature representations extracted by different pre- trained CNN models. To our knowledge, there has been no previous study on using features extracted from Convolutional Neural Networks (CNNs) Models and Support Vector Machine (SVM) classifier to classify Street View images in the context of image geo-localization problem. The purpose of this study is to enhance the prediction of location from photos captured in urban settings by approaching it as a classification problem. For assessment purposes, we utilized the Google Street View (GSV) dataset consisting of high-resolution images captured in downtown and neighboring areas of Orlando, Pittsburgh, and a portion of Manhattan. Every Street View placemark (i.e. each spot on one street), there are four (4) side scenes and one (1) upward view. In addition, each location is accompanied by an extra image, displaying overlaid markers like the address and street names.

Thesis Outline

The remainder of this thesis is structured in four chapters as follows:

- The first chapter presents the basic concepts of image analysis and computer vision. Then, the phases of feature extraction and classification and their approaches studied in the literature are explored. Finally, we discuss the difficulty of computer vision.
- The second chapter presents the definitions and principal concepts of artificial intelligence, machine learning and deep learning. Then, the relation between machine learning and deep learning is also explained. Subsequently, we review some types of deep learning networks, the CNN structure, and the most frequent methods for object classification using deep learning. Finally, common CNN model architectures are mentioned.
- The third chapter presents the general concepts and reviews the prominent approaches in the field of image geo-localization. Finally, different existing techniques and the methods introduced in the literature are cited.
- The fourth chapter presents our contribution to enhancing Street View Image classification and the obtained results. Starting by outlining the overall architecture of the proposed system and then studying each of its two main phases separately. Finally, experimental design, analyze the results, and interpret their significance are presented.

At last, a summary, perspectives and future directions are given in conclusion.

CHAPTER 1

IMAGE ANALYSIS AND COMPUTER VISION

CHAPTER 1: IMAGE ANALYSIS AND COMPUTER VISION

1.1. Introduction

Image analysis and interpretation are the main tools of computer vision, tackling problems that humans solve unknowingly. Computer vision aims to develop machines that reconstruct and interpret a three-dimensional environment based on radiant energy measurements. It is the theory underlying the capacity of artificial intelligence systems to view and recognize their surrounding environment.

In this chapter, we will first present basic concepts of image analysis and computer vision. Then, we will explore the phases of feature extraction and classification and their approaches studied in the literature. Finally, we will discuss the difficult of computer vision.

1.2. Definition of image analysis and computer vision

Using the machine for image interpretation may be considered as an effort to discern relationships between the input imagery and previously established models representing the observable physical environment. The transition from the input image to the model reduces the information contained in the image to information that is relevant to the application domain (Sonka, Hlavac, & Boyle, 2015). This process is generally divided into numerous steps and numerous levels representing the image are used. The lower level includes raw image data while the higher levels interpret it. Computer vision designs these intermediate representations and algorithms that are used to establish and maintain relationships between entities within and between layers.

According to data organization, image representation may be divided into four levels as shown in Figure 1.1, which depicts a global image interpretation system where the image analysis is carried out from the signals to the abstract description needed for the understanding of an image.

To automate the corresponding tasks carried out by biological vision systems and perhaps even extend some of their capabilities, computer vision aims to create machines that reconstruct and interpret a three-dimensional environment based on measurements of radiant energy (Council, 1991).

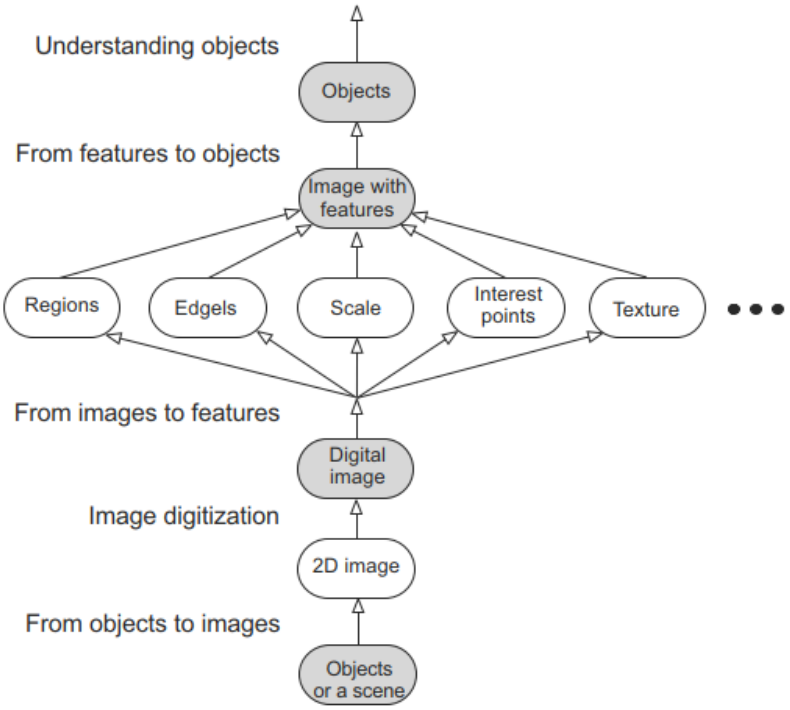

Figure 1.1. Levels of image representation suitable for image analysis problems (Sonka et al., 2015)

The representations can be grouped into four levels, although the boundaries between them are not rigid, and a more nuanced classification of representation levels is employed in certain applications. These four levels of representation span from low-level signals to high-level descriptions that humans can comprehend. The flow of information between these levels can occur bi-directionally. The first level is the lowest level of iconic image representation; it consists of the image's original data: integer matrices corresponding to the luminous intensity of the image's image pixels. This image data typically represents the image capture device direct output, such as a scanner.

The information in the image matrix can be accessed through the coordinates of a pixel, which correspond to the row and column indices. The matrix serves as a complete representation of the image, implicitly containing spatial relationships between semantically important parts. In the case of an image, space is two-dimensional, and a natural spatial relationship within a matrix is a neighborhood relationship. Programming languages commonly employ an array-based data structure to model matrices, and contemporary hardware typically has sufficient physical memory to support this data structure for images. In cases where there are limitations with physical memory, virtual memory can often be used transparently. In the past, the constraints of memory capacity posed a substantial obstacle to the development of image-based applications.

The matrix contains a large amount of image data, leading to slow processing. Accelerating the algorithms is possible by deriving global information from the original image data, resulting in more concise information that necessitates less memory.

Segmented images comprise the second level of representation. The image components are clustered into groups, which likely correspond to the same objects. Knowledge about the application context can facilitate image segmentation and aid in recognizing the implications of image noise.

The third level of representation comprises geometric models that embody knowledge about (2D) and (3D) shapes. Quantifying these geometric properties is highly challenging, yet also imperative. Geometric representations prove beneficial when conducting extensive and intricate simulations to analyze the influence of lighting and motion on physical entities.

Relational models constitute the fourth level of image data representation. These models enable us to process data more efficiently and at a more abstract level. All techniques are often explored; semantic frames can represent the information obtained from the image.

1.2.1. Image analysis

Image analysis, also known as image understanding, lies between computer vision and image processing domains. An image is characterized as a two-dimensional function, $f(x, y)$, with x and y representing spatial coordinates and the value of f at any pair of coordinates (x, y) denoting the intensity or gray level of the image at that specific point. If x, y, and the intensity values of f are all finite discrete quantities, then we refer to it as a digital image. Digital image processing involves manipulating digital images using a digital computer. It's important to note that a digital image comprises a finite number of elements; each element has its location and value which can be referred to in various ways such as picture elements (pels) or pixels (Gonzalez & Woods, 2018).

1.2.2. Computer vision

Computer vision is addressed exceedingly complex challenges, with the objective of achieving analogous outcomes to those delivered by natural visual systems (Sonka et al., 2015). It is the study of how computers see and understand digital images and videos. It is the theory underlying the capacity of artificial intelligence systems to view and recognize their surrounding environment.

The promising reality is that computer vision technology has found widespread application and is presently being utilized across a diverse array of practical, real-world domains, which include handwritten recognition, medical imaging, self-driving vehicles, fingerprint recognition and biometrics ...etc. (Szeliski, 2022).

1.3. Feature extraction

Extraction of feature addresses the challenge of identifying the most pertinent collection of features and providing useful information to increase the effectiveness of processing and data storage. Feature vectors remain the most prevalent and practical approach to data representation for regression and classification tasks. Then, data may be organized in simple tables; where lines denoting “entries”, “data points”, “samples”, or “patterns”, and columns denoting “features”. Features are derived from quantitative or qualitative measurements and can be considered as “attribute” or “variable”. The current approach to feature extraction is influenced by the growing size of data tables, which is enabled by advancements in data storage technology (Isabelle Guyon, Steve Gunn, Masoud Nikravesh, 2006).

The feature extraction phase represents a critical and main step in classification since the performance of classifications is highly dependent on the descriptors used and is relative to the quality of the descriptor extraction techniques used (Monika Bansal, Kumar, Sachdeva, & Mittal, 2021). This step involves automatically extracting different image characteristics to get descriptors, often known as a vector of characteristics or signatures. The selected descriptor should be the most representative, discriminating, and fastest in terms of computation time. A good descriptor should be able to describe the image content with certain variations, including illumination changes, the variation of the scale, the translation and rotation, the change of viewpoint, etc. Generally, we can divide the existing feature extraction methods into three main categories depending on the extracted features: low-level feature-based methods, mid-level feature-based methods, and high-level feature-based methods. Therefore, features can be categorized into different levels; low-level features like points, corners, and lines, mid-level features like blobs and planes, and high-level features like objects as illustrated in Figure 1.2.

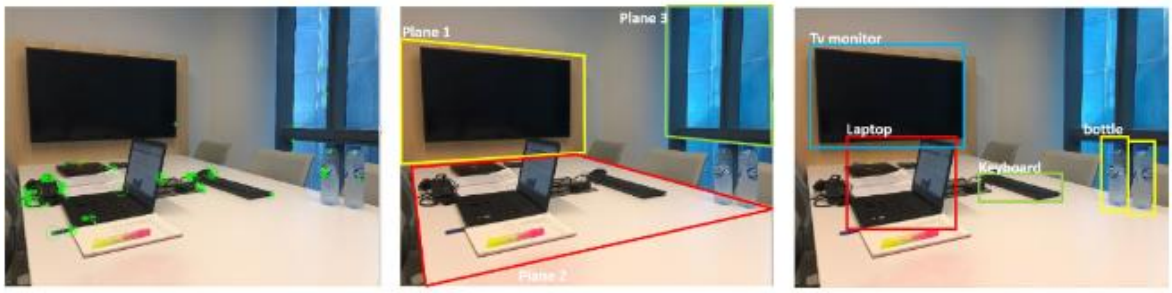

Figure 1.2. Different features are extracted from the same frame. Left: low-level features (SURF), middle: mid-level features (planes), right: high-level features (objects)

1.3.1. Low-level feature-based methods

Methods based on low-level features or visual features are interested in the information contained within the image at the pixel level, such as color, texture, shape, information spatial and spectral, or their combinations, that constitute the main characteristic of an image and therefore carry useful information for classification. Descriptors characterizing the whole image (global descriptor) or several local descriptors each characterizing a part of the image can be used. However, local descriptors are typically preferred over global ones in modern imaging techniques because they are more effective, allow a finer search, and better absorb certain variations. Therefore, there are two main strategies for extracting visual features: global feature extraction approaches, which compute descriptors using the entire data in the image and also can generalize the visual content with a single feature vector; and local feature extraction approaches, which compute descriptors on edges between regions or points of interest (Andrade, Almeida, Pedrini, & Torres, 2012).

a. Global feature extraction approaches

In the case of providing observations on the whole image; the extraction of global descriptors makes it possible to extract a single descriptor, which describes the entire information of the visual content, that makes them robust to noise (Andrade et al., 2012).

These characteristics enable us to reduce the number of calculations required, the size of the database, and the cost of classification. However, the global approach does not allow for the differentiation of image parts; they do not distinguish, for example, objects in the image, unless the image contains only one object against a plain background. Image feature descriptors are generated by encoding the color, texture, and shape data of an image into vector representations that capture the visual characteristics of the image (S. Wang, Han, & Jin, 2019).

Color descriptors

Color is one of the main and effective low-level features that allow humans to recognize images. It is a property that is dependent on light reflection to the eye as well as brain processing of that information. Color is commonly used to distinguish certain objects, places, and times of day (X. Wang, Zhang, & Yang, 2014)(R. Sharma & Singh, 2014).

Color is a consistent and resilient global feature that is more reliable than other image characteristics. It considered as the most obvious and intuitive element of an image, and it is resistant to changes in image size, noise, resolution and orientation (Alzubi, Amira, & Ramzan, 2015)(Dewan & Thepade, 2020).

In general, spaces of color are divided to linear color spaces such as RGB, XYZ, CMY, YIQ, YUV, and non-linear color spaces like L*a*b, HSV, Nrgb, Nxyz, L*u*v (X. Wang et al., 2014). The RGB color space is a three-color additive color space comprised of blue, green, and red. Secondary colors could be created by combining primary colors; for example, combining blue and red produces magenta, blue and green produces cyan, and a full-intensity combination of blue, green, and red produces white. Yet, the RGB space is inefficient in dealing with real-world images, so it is avoided in most image retrieval algorithms due to its inability to measure perceptual similarity. Furthermore, in terms of human visual perception, the distances in RGB space have little meaning (Alzubi et al., 2015).

As a result, the color space of HSV is used in place of the RGB color space because the components of saturation and hue closely resemble the perception of human visual. The HSV model has three components: the hue, that refers to the color, the saturation, that refers to the color's vibrancy, and the value, that refers to the color's brightness. The YCbCr color space is divided into luminance (Y) and chrominance (Cb, Cr), with Cb denoting the blue-yellow color difference and Cr denoting the red-green color difference, respectively. To achieve perceptual uniformity, the L*a*b* color space is also derived from the XYZ color space. Based on the color opponent process, L*a*b has one lightness dimension (L) and two chrominance dimensions (a*, b*)(Alzubi et al., 2015).

The most commonly used color representations are described below:

- Color Histograms: considered as the most widely utilized description method for color feature, first suggested by Ballard and Swain in 1991 (Swain & Ballard, 1991) for indexing

and retrieving images based on content and color. The goal is to calculate pixels in common between the two histograms number. This metric was developed for finding images that contain a specific object, which is represented by the query image.

They show the color information distribution in images with scaling invariance, rotation and translation. Nonetheless, storage costs and high computational are unavoidable flaws of the classic color histogram.

- Color Correlogram: expresses how the spatial correlation of two colors changes with distance (R. Sharma & Singh, 2014). It only captures the color distribution in an image and excludes any spatial correlation information.

Because color histograms lack information about color spatial distribution, different representations, such as color correlograms and auto-correlograms, have been proposed. These techniques give information about color pairs changes spatial correlation with distance in the image, as well as they outperform color histograms in retrieval performance (J. Huang, Kumar, Mitra, Zhu, & Zabih, 1997).

Notation: Let I be an $n \times n$ image. (We assume the image is square, for simplicity). The colors in I are quantized into m colors $c_1, \ldots, c_m$. (In practice, m is considered to be a constant and hence we eliminate it from our running time analysis).

For a pixel $p = (x, y) \in I$, let $I(p)$ denote its color. Let $I_c \triangleq \{p \mid I(p) = c\}$. Thus, the notation $p \in I_c$ is synonymous with $p \in I, I(p) = c$. For convenience, we use the L_∞-norm to measure the distance between pixels, for example, for $p_1 = (x_1, y_1), p_2 = (x_2, y_2)$, we define $|p_1 - p_2| \triangleq \max \{|x_1 - x_2|, |y_1 - y_2|\}$. We denote the set *{1,2,….,n}* by *[n]*.

Definition: The histogram h of I is defined for $i \in [m]$ by

$$h_{c_i}(I) \triangleq n^2 . \; Pr_{p \in I}\left[p \in I_{c_i}\right] \qquad (1)$$

For any pixel in the image, $h_{c_i}(I) / n^2$ gives the probability that the color of the pixel is c_i.

Let a distance $d \in [n]$ be fixed a priori. Then, the correlogram of I is defined for $i, j \in [m], k \in [d]$ as

$$\gamma_{c_i,c_j}^{(k)}(I) \triangleq Pr_{p_1 \in I_{c_i}, p_2 \in I}\left[p_2 \in I_{c_j} \middle| |p_1 - p_2| = k\right] \qquad (2)$$

Given any pixel of color c_i in the image, $\gamma_{c_i,c_j}^{(k)}$ gives the probability that a pixel at a distance k away from the given pixel is of color c_j. Note that, the size of the correlogram is $O(m^2 d)$. The auto-correlogram of I captures the spatial correlation between identical colors only and is defined by

$$\alpha_c^{(k)}(I) \triangleq \gamma_{c,c}^{(k)}(I) \qquad (3)$$

This information is a subset of the correlogram and requires only $O(md)$ space.

We must address the following issue when choosing *d* to define the correlogram. A large *d* would necessitate expensive computation and storage. Therefore, a small *d* could degrade the quality of the feature.

- Color Moments: are a straightforward and efficient color characterization algorithm, that can be used to measure the similarity between two images (Dewan & Thepade, 2020)(Stricker & Orengo, 1995). The skewness, standard deviation and mean are utilized to depict the color offset, color variance and average color, that are enough to characterize the images color distribution (D. P. Tian, 2013). The following are the mathematical expressions for diffrent moments of color:

$$\mu_i = \frac{1}{n}\sum_{j=1}^{n} h_{ij} \qquad (4)$$

$$\sigma_i = \left[\frac{1}{n}\sum_{j=1}^{n}\left(h_{ij} - \mu_i\right)^2\right]^{1/2} \qquad (5)$$

$$\gamma_i = \left[\frac{1}{n}\sum_{j=1}^{n}\left(h_{ij} - \mu_i\right)^3\right]^{1/3} \qquad (6)$$

Where h_{ij} is the frequency at which pixels in the *i*- th color channel component of the gray image have a gray level of *j*. Every color component in an image has three low-order color moments.

As a result, the components of three- color *Y, U, V* in this image can be used to create a 9- dimensional histogram vector composed of the image's first three- color moments. Thereafter, the image color characteristics can be expressed as:

$$F = [\mu_Y, \sigma_Y, \gamma_Y, \mu_U, \sigma_U, \gamma_U, \mu_V, \sigma_V, \gamma_V] \quad (7)$$

Color moments are typically utilized for the first filtering to minimize the range of search by deleting images that are not identical in color. It only requires 9 feature vectors as color features to benefit the need for quantization when calculating and color moments feature small dimension. The color moments in the CBIR that characterize an image feature not increase only the retrieval efficiency but also ensure image color integrity in retrieved images (S. Wang et al., 2019).

- Color Coherence Vector: represents a development from the Color Histogram method, aimed at discerning the image color characteristics. Essentially, within every histogram group, pixels are divided into two parts: coherent and incoherent constituents (Pass & Zabih, 1996). If some pixels occupied continuous area within the category is larger than a specified threshold, these pixels could be identified as coherent, as well as this area is recognized as a connected area. Alternatively, if these pixels do not meet this condition, they are considered incoherent pixels, and the corresponding area is regarded as disconnected (Dewan & Thepade, 2020).

Because the Color Coherence Vector also includes spatial information, it often exhibits superior performance compared to a Color Histogram. However, the dimension of a Color Coherence Vector is significantly greater than that of a traditional histogram, typically around twice the size (D. Zhang, Islam, & Lu, 2012).

The relationship between the threshold τ of the image and the number of pixels *N* is determined as $\tau = 0.01\ N$. By choosing α_i to describe the number of *i*- th coherent pixels of the Color Histogram and β_i to characterize the number of *i*- th incoherent pixels of the Color Histogram (C. Huang & Wang, 2006). Therefore, the Color Coherence Vector can be expressed as:

$$\{(\alpha_1, \beta_1), (\alpha_2, \beta_2), \ldots, (\alpha_n, \beta_n)\} \quad (8)$$

- Color sets: can be considered as an approximation of the Color Histogram (Smith & Chang, 1996). It indicates the binary vector which could be described by B^M, an *m*-dimensional binary space in that each space axis corresponds to unique search *m*.

When the color *m* appears, color Sets as a 2D vector in the B^M space corresponds to the selection of color, it is known that *c[m] =1*; contrary, *c[m] =0*. These Color Sets may directly be generated by adjusting the Color Histogram threshold as follows:

$$c[m] = \begin{cases} 1, h[m] \geq \tau_m \\ 0, h[m] < \tau_m \end{cases} \quad (9)$$

Where *c[m]* is the Color Sets, *h[m]* is the Color Histogram, and $\tau[m]$ represents the color threshold for color *m*.

Color Sets are realized as follows: (r, g, b) represent the three- color point group of the three- dimensional RGB color space, *(x, y, z)* is the three- color point group of the general color space *XYZ*, and *T* is the conversion relationship between the RGB color space and the *XYZ* color space (S. Wang et al., 2019).

Defining Q_M as a function of quantization that quantizes *(x, y, z)* into *M* groups; the color search set for *(x, y, z)* can then be obtained as follows:

$$(x, y, z) = T(r, g, b) \quad (10)$$

$$m = Q_M(x, y, z) \quad (11)$$

Where *m = 0, 1,…, M-1*.

- Dominant color: is considered as one of MPEG-7 color descriptors and it is widely applied in image retrieval. It describes the distribution of the prominent colors in the image. In contrast to the bin quantization method employed in histograms, the specification of colors within a dominant color descriptor is solely constrained by the extent to which the color space has been quantized. It aims to provide an efficient, compact, and intuitive representation of colors found within a region of interest or an image (Manjunath, Ohm, Vasudevan, & Yamada, 2001)(Shao, Wu, Cui, & Zhang, 2008).

To compute this descriptor, the colors in a given image or region are first clustered. This yields a small number of colors and the percentages of which are calculated. The variances of the colors assigned to a given dominant color are also computed as an option. The color percentages in the region should add up to 1. A spatial coherency value is also computed, which distinguishes between large color blobs and colors that are spread throughout the image. As a result, the descriptor is defined by:

$$F = \{\{c_i, p_i, v_i\}\, s\}, (i = 1,2, \dots, N) \quad (12)$$

Where

c_i *i*th dominant color;

p_i its percentage value;

v_i its color variance;

s is a single number that indicates the overall spatial homogeneity of the dominant color in the image;

N is the number of dominant color;

When the color variance *V* and the spatial uniformity of the dominant color *S* are not taken into account, the expression above can be simplified to:

$$F = \{c_i, p_i\}, (i = 1,2, \dots, N),\ p_i \in [0,1] \quad (13)$$

The number of dominant colors *N* varies between images, with a maximum of eight dominant colors used to represent the region. The color variance is an optional field (Manjunath et al., 2001). Each percentage value p_i is quantized to five bits. Color quantization is determined by the color space specifications defined for the entire database and is not required for each descriptor.

The described method (Leszek Cieplinski, Kim, Ohm, Pickering, & Yamada, 2001) to extract the dominant color is based on using the generalized Lloyd algorithm for color clustering. This problem is formulated as one of minimizing the distortion D_i in each cluster i

$$D_i = \sum_n v(n)\|x(n) - c_i\|^2 \qquad x(n) \in C_i \quad (14)$$

Where

c_i is the centroid of cluster C_i;

x(n) is the color verctor at pixel;

v(n) is the perceptual weight for a pixel *n.*

Perceptual weights are calculated from local pixel statistics to consider for the fact that human vision is more sensitive to detecting changes in smooth areas compared to textured ones. These specific perceptual weights can be found in the reference (Leszek Cieplinski et al., 2001). Next, calculation and quantization color variance of the colors within cluster Ci, including the dominant color ci, which is represented with a 3-bit quantization for each color variance (Manjunath et al., 2001).

- Color co-occurrence matrix: the conventional color co-occurrence matrix is structured as a three-dimensional matrix, where the colors of any given pair are arranged along the first and second dimensions, while the spatial distance between them is positioned along the third dimension (Kovalev & Volmer, 1998). In this context, the conventional Color Co-occurrence Matrix (CCM) is equivalent to a Color Correlogram (J. Huang et al., 1997). The Color Co-occurrence Matrix (CCM) is simplified to represent the number of color (hue) pairs between neighboring (adjacent) pixels in the image. This simplification considers the four neighboring pixels (both horizontal and vertical neighbors) for each pixel in the image (Chang, Pyun, Ahmad, Chun, & Park, 2005).

Let *I* be an image *(N×M)*, quantized into *m* colors and *p(x, y)* represents the color of the image pixel *(x, y).* Then, the simplified Color Co-occurrence Matrix (CCM) is defined as follows:

$$H^I(i,j) = \eta\left(\left(p(x,y), p\left(N_{(x,y)}\right)\right) == (i,j)\right)$$

$$= \alpha \sum_{x=1}^{N} \sum_{y=1}^{M} C_i(x,y) \quad \sum_{(x',y')\epsilon N_{(x,y)}} C_j(x',y') \qquad (15)$$

Where η indicates the number of times $\left(p(x,y), p\left(N_{(x,y)}\right)\right)$ equals the value of the color indices *(i,j)*, and $N_{(x,y)}$ indicates 4-neighbors of the pixel *(x,y).* $C_i(x,y)$ is given by

$$C_i(x,y) = \begin{cases} 1 & if\ p(x,y) = i \\ 0 & o.w \end{cases} \qquad (16)$$

The normalization constant α is $1/4 \cdot N \cdot M$, for the total number of pixel pairs $\left(p(x,y), p\left(N_{(x,y)}\right)\right)$ is approximately $4 \cdot N \cdot M$ by discounting the difference of boundary pixels.

The reason the simplified Color Co-occurrence Matrix (CCM) is symmetric because the adjacent pixel pairs are neighbors of each other. Color quantization was performed with 16 colors, as it has been empirically demonstrated that 16 colors (in the hue model) provide adequate accuracy for color-invariant object retrieval. As a result, the dimensions of the simplified Color Co-occurrence Matrix are 16×16 (Shim & Choi, 2003).

Texture descriptors

Similar to color, texture serves as a robust low-level descriptor for applications involving image search and retrieval (Manjunath et al., 2001). Texture represents another significant image attribute. Unlike color, which is typically associated with individual pixels, the texture is a characteristic that can only be assessed using a cluster of pixels. Thanks to its pronounced discriminative potential, texture features find extensive application in image retrieval and semantic learning methodologies. Texture has been thoroughly investigated within the fields of image processing and computer vision (Tamura, Mori, & Yamawaki, 1978).

Representing texture is a complex task. It involves primarily modeling texture as a two-dimensional variation in gray levels within an image. This modeling calculates the relative brightness between pairs of pixels, allowing us to estimate factors like contrast, regularity, coarseness, and directionality. However, the challenge lies in discerning patterns of co-variation among pixels and connecting them to specific texture categories, such as silky or rough (Tamura et al., 1978)(Chi Zhang & Huang, 2014).

Several methods have been suggested for extracting texture features. These methods can be broadly categorized into two groups based on the domain from which the texture feature is derived: spatial texture feature extraction methods and spectral texture feature extraction methods. In the sections below, we will elaborate on these techniques.

-Spatial texture feature extraction methods: in the spatial approach, texture features are obtained by either calculating pixel statistics or identifying local pixel structures within the original image domain. Spatial texture feature extraction techniques can be further categorized into structural, statistical, and model-based methods (D. Zhang et al., 2012).

- *Structural methods* identify the fundamental structures within an image and determine their locations. These approaches prove valuable when dealing with images featuring highly regular textures and are particularly effective for images

containing human-made objects characterized by consistent patterns (Dewan & Thepade, 2020). It characterizes textures through a collection of texture primitives (referred to as texons or texture elements) and their arrangement rules (Fu, 1977; Gonzalez & Woods, 2018; Materka & Strzelecki, 1998). These texons are structured into a string descriptor, and syntactic pattern recognition techniques are employed to assess the similarity between two such descriptors.

- *Statistical methods* are straightforward and commonly applied techniques that utilize quantitative measurements of intensity distributions within a region (Dewan & Thepade, 2020; S. Wang et al., 2019). The statistical texture features quantify texture by evaluating the low-level statistical properties of grayscale images. Common statistical features in the spatial domain include moments (Gonzalez & Woods, 2018; F. Long, Zhang, & Feng, 2003), Tamura texture features (M. M. Islam, Zhang, & Lu, 2008; Tamura et al., 1978; Yavlinsky, Schofield, & Stefan, 2005) and features derived from the Grey Level Co-occurrence Matrix (GLCM) (F. Long et al., 2003; Park, Lee, & Kim, 2004). Statistical features are both compact and robust due to their derivation from extensive data support. Nevertheless, they may not be comprehensive enough to effectively describe the large variety of textures.
- In *model-based methods*, texture is understood through the application of stochastic (random) or generative models. Model parameters serve to define the intrinsic texture attributes present in the image. The model-based methods use the parameters of model to represent texture features, and characterize not just the texture locality randomness but also the texture regularity all at once (S. Wang et al., 2019). Popular texture models encompass the Markov random field (MRF) (Cross & Jain, 1983; Liu & Picard, 1996; F. Long et al., 2003; Luo & Savakis, 2001; Materka & Strzelecki, 1998; TUCERYAN & JAIN, 1993; Vailaya, Figueiredo, Jain, & Zhang, 2001), the simultaneous auto-regressive (SAR) model (J. Z. Wang, Li, & Wiederhold, 2001), and fractal dimension (FD) (Chaudhuri & Sarkar, 1995; Materka & Strzelecki, 1998), among others. Since these models often entail optimization processes, they tend to be computationally demanding.

Summary: Spatial texture feature extraction methods are simple to understand, and many of them include semantics. They do not require a regular region shape and can be easily applied to irregular regions. These features, however, are usually sensitive to noise and

distortions. Moreover, many of these methods entail complex search and optimization processes with no general solutions.

-Spectral texture feature extraction methods: in the spectral approach, an image is converted into the frequency domain, after which features are computed based on the transformed image. Fourier transform (FT) (Hervé & Boujemaa, 2007; Lee & Chen, 2005) is one of the most commonly used spectral techniques, also the discrete cosine transform (DCT) (Z.-M. Lu, Li, & Burkhardt, 2006), wavelet (Fan, Gao, Luo, & Xu, 2004; Park et al., 2004; J. Z. Wang et al., 2001), and Gabor filters (Manjunath & Ma, 1996; Salembier & Manjunath, 2002; D. Zhang, Wong, Indrawan-Santiago, & Lu, 2000; R. Zhang, Zhang, Li, Ma, & Zhang, 2005).

Although the Fourier transform (FT) and discrete cosine transform (DCT) are very fast, they are not scale and rotation invariant. Wavelet is efficient and robust, but it only captures horizontal and vertical features. Among these methods, Gabor features are the most robust because they capture image features across multiple orientations and scales. In recent studies focusing on multi-resolution analysis, it has been demonstrated (Sumana, Islam, Zhang, & Lu, 2008) that curvelet features offer notable advantages compared to Gabor and wavelet features. This is attributed to the greater effectiveness of curvelet features in capturing curvilinear properties, such as lines and edges (Starck, Candès, & Donoho, 2002).

Shape descriptors

Shape descriptors are also counted as an important low-level feature due to their utility in the identification of real-world shapes and objects (Latif et al., 2019). It describes image content which is an essential problem in computer vision and pattern recognition.

In the context of image retrieval, the necessity for shape representation to be invariant to translation, rotation, and scaling varies depending on the specific applications. Clearly, if a representation meets the requirements for invariance to translation, rotation, and scaling, it automatically satisfies the requirements for invariance to other transformations. Consequently, in the subsequent discussion, our emphasis will be on shape representations that exhibit transformation invariance (Rui & Huang, 1999).

Extraction of features based on shape and description techniques is generally divided into two main categories: region-based and contour-based techniques. The region-based technique uses all the pixel values of the object and extracts features from the entire region,

while contour-based techniques are those which use the information and calculate the shape features only from the boundary of an object. Contour-based techniques are classified as complete object shape-based when they represent the entire boundary as a single shape, and they are considered primitive or structure-based when the boundary is segmented into parts and then described. Moreover, these techniques, as they focus on a part of the region, tend to be more sensitive to noise compared to region-based techniques. Small alterations in the shape can have a notable impact on the shape contour. As a result, color image retrieval typically relies on region-based shape features (Dewan & Thepade, 2020; D. Zhang et al., 2012). Region-based techniques are categorized into two main types: spatial domain-based and transform domain-based. Within the spatial domain-based techniques, there is a further distinction between complete object-based and primitive-based methods, depending on which aspect of the object is being described (Dewan & Thepade, 2020).

- **Fourier descriptors:** (FDs) are a robust feature for representing various boundaries and objects (R. Sharma & Singh, 2014). They are widely used shape feature descriptors known for their attributes such as computational efficiency, clarity, and the capacity to provide a gradual description of shape characteristics from coarse to fine. These descriptors capture the shape features of an image by performing a Fourier transformation on the object's boundary (S. Wang et al., 2019).

Suppose we have an N-point digital boundary that initiates from an arbitrary point (x_0, y_0) and consistently follows a counterclockwise direction along the boundary. Then, we can generate a set of coordinate pairs (x_0, y_0), $(x_1, y_1), \ldots, (x_{N-1}, y_{N-1})$. These coordinates can be represented in a complex form, such as $z(n) = x(n) + jy(n), n = 0,1,2,\ldots,N-1$. The given expression represents the Discrete Fourier Transform (DFT) of $z(n)$ as:

$$a(k) = \sum_{n=0}^{N-1} z(n)\, exp\left[\frac{-j2\pi kn}{N}\right], 0 \le k \le N-1 \qquad (17)$$

Where $a(k)$ the complex coefficients are named the Fourier Descriptors of the boundary.

- **Hough transform:** (HT), introduced by Hough (Paul V C Hough, 1960), is a widely employed method for distinguishing geometric shapes that share certain features within an image. This technique is particularly famous for identifying lines that appear at varying angles. The Hough Transform exhibits robustness against interference, making it less affected by issues such as noise, missing segments of lines within the image and the presence of different non- linear structures when detecting straight lines (S. Wang et al., 2019).

- **Invariant moments: (**IMs) which are depend on algebraic invariants, they are introduced by Hu (Hu, 1962), who utilized the grayscale distribution moments of an image to characterize its grayscale feature distribution. For a digital discrete image represented as *f(x, y)*, the definition of the *p+q* moment is as follows:

$$\mu_{pq} = \sum_x \sum_y (x - \bar{x})^p (y - \bar{y})^q f(x, y) \tag{18}$$

Where *x* and *y* represent the positions of pixel in the image, *f (x, y)* denotes intensity of pixel and the *p+q* order central moment is defined as:

$$\mu_{pq} = \sum_x \sum_y (x - \bar{x})(y - \bar{y})^q f(x, y) \tag{19}$$

$$\eta_{pq} = \frac{\mu_{pq}}{\mu_{00}^{\gamma}}$$

Where $\bar{x} = m_{10}/m_{00}$, $\bar{y} = m_{01}/m_{00}$.

Then, it can be normalized the the *p+q* order center moment as follows:

$$\eta_{pq} = \frac{\mu_{pq}}{\mu_{00}^{\gamma}} \tag{20}$$

In which $\gamma = (p + q)/2,\ p + q = 2,3, \dots, n$. Hence, we can generate seven sets of moments based on the normalized center distance, as follows:

$$\varphi_1 = \eta_{20} + \eta_{02}$$

$$\varphi_2 = (\eta_{20} - \eta_{02})^2 + 4\eta_{11}^2$$

$$\varphi_3 = (\eta_{30} - 3\eta_{12})^2 + (3\eta_{21} - \eta_{03})^2$$

$$\varphi_4 = (\eta_{30} + \eta_{12})^2 + (\eta_{21} + \eta_{03})^2$$

$$\varphi_5 = (\eta_{30} - 3\eta_{12})(\eta_{30} + \eta_{12})\,[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] + (3\eta_{21} - \eta_{03})(\eta_{21} + \eta_{03})\,[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2]$$

$$\varphi_6 = (\eta_{20} - \eta_{02})[(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2] + 4\eta_{11}(\eta_{30} + \eta_{12})(\eta_{21} + \eta_{03})$$

$$\varphi_7 = (3\eta_{21} - \eta_{03})(\eta_{30} + \eta_{12})\,[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] - (3\eta_{21} - \eta_{03})(\eta_{21} + \eta_{03})[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2]$$

(21)

- **Zernike moments:** Zernike introduced in 1934 a collection of orthogonal polynomials defined on the circle unit, known as the Zernike orthogonal polynomials, in the following manner:

$$R_{nm}(r) = \sum_{s=0}^{\frac{n-|m|}{2}} \frac{(-1)^s (n-s)!}{s!(1/2(n+|m|)-s)!(1/2(n-|m|)-s)!} r^{n-2s} \quad (22)$$

Where *n* is a non-negative integer, and *m*, *n* are the orders of the Zernike polynomial.

The Zernike moment descriptor (R. Sharma & Singh, 2014) is widely regarded as one of the most appropriate choices for shape-based retrieval. It is valued for its robustness, computational efficiency, retrieval performance, and compact representation. Additionally, orthogonal moments, including Zernike moments, offer certain advantages such as increased resilience in the presence of image noise. The benefits of Zernike moments include robustness to any minor variations in shape and noise, rotation invariance property, effectiveness when the image can be characterized in a better way by a small set of its Zernike moments instead of any other types of moments like the geometric moments, expressiveness is enhanced due to the orthogonality of the basis which resulting in minimal information redundancy, and multilevel representation while the global shape of a pattern can be described by a compact set of Zernike moments. Within this set, the lower-order moments capture the overall pattern shape, while the higher-order moments convey finer details.

b. Local feature extraction approaches

Methods based on local features extract low-level characteristics from key points or prominent patches within an image. Hence, local descriptors have demonstrated their effectiveness in enhancing the performance of Content-Based Image Retrieval (CBIR) systems.

Local feature extraction techniques are valuable for the detection of human-made objects within images (Dewan & Thepade, 2020). These techniques define salient regions, often referred to as interest points or key points, and then describe the neighborhood patches surrounding these key points to characterize the image. The identified key points must be highly repeatable, ensuring their detection remains consistent under various transformations like rotation and changes in illumination. Effective local features should exhibit the

following properties: distinctiveness (displaying high variations), locality (remaining sufficiently small to prevent occlusion from different viewing angles), quantity (ensuring a sufficiently large number of features for matching purposes), accuracy (precise identification across various scales), efficiency (fast detection), invariance to substantial deformations, and robustness to minor deformations (Tuytelaars & Mikolajczyk, 2007).

There are many methods for local feature detection techniques, and some of the most commonly used ones are:

- **Harris Detector**: this method uses the edge and corner detector based on the autocorrelation function (Harris & Stephens, 1988). It computes intensity differences in all directions for the shift of *(u, v)* based on the following equation:

$$E(p,q) = \sum_{u,v} \omega(u,v)[I(p+u, y+q) - I(u,v)]^2 \quad (23)$$

The window function is given as:

$$\omega(u,v) = e^{-(u^2+v^2)/2\sigma^2} \quad (24)$$

E(p,q) for small changes shift *(p,q)* can be expressed as:

$$E(p,q) = (p,q)M\ (p,q)^T \quad (25)$$

Then, *M* is given as:

$$M = \sum_{p,q} \omega(p,q) \begin{bmatrix} IpIp & IpIq \\ IqIp & IqIq \end{bmatrix} \quad (26)$$

Where *Ip* and *Iq* are the gradients in *p* and *q* direction respectively; α and β are Eigenvalues of matrix *M*; then, trace of *M* is $\alpha+\beta$, and the determinant of *M* is $\alpha\beta$.

The region is a corner region if CR has a positive value:

$$CR = (\alpha.\beta) - k.(\alpha+\beta)^2 \quad (27)$$

- **Shi and Tomasi Corner Detection**: this detector operates at a higher speed compared to the Harris corner detector to achieve better corner detection. It follows a similar approach as the Harris corner detector, with minor adjustments in the conditions used to identify a region as a corner. Specifically, when the corner region (CR) value exceeds the threshold, the region is marked as a corner (Shi & Tomasi, 1994).

$$CR = min(\alpha, \beta) \quad (28)$$

- **Scale-Invariant Feature Transform (SIFT)**: is a prominent floating-point descriptor and it is among the most widely used local features, known for its outstanding performance (Lowe, 2004). The SIFT algorithm can be divided into four primary stages: scale-space extrema detection, keypoint localization, orientation assignment, and keypoint descriptor. It detects recurring features in the image, which remain recognizable across different scales and views by using a scale-space function that leverages the difference of the Gaussian function applied to two adjacent scale-separated images (Dewan & Thepade, 2020). Then, it generates a 128-dimensional feature vector for each keypoint. This feature vector maintains invariance to changes in image scale and rotation. Furthermore, it demonstrates robustness against a significant range of affine distortions, noise, and specific levels of illumination changes. Nonetheless, when an image contains numerous keypoints, this leads to the creation of high-dimensional descriptors, particularly in the context of large-scale image retrieval (Niu, Zhao, Lin, & Zhang, 2020).

- **Speeded-Up Robust Features (SURF)**: is an efficient alternative to the SIFT method. It relies on the determinant value of the Hessian matrix to detect feature points and expedites operations through the use of integral graphs. The SURF, built on the Hessian matrix, offers an approximation of the Laplacian of Gaussian (LoG) by utilizing integral images and the ability to operate across multiple scales simultaneously. It relies on the determinant of the Hessian matrix to compute keypoint location and scale. Additionally, it employs Haar wavelet responses for orientation assignment and description, which results in a 64-dimensional vector. It's important to note that SURF does not possess affine invariance (Dewan & Thepade, 2020).

- **Features from Accelerated Segment Test (FAST)**: is one of the high-speed machine learning-based corner detection techniques (Rosten & Drummond, 2006). It employs a segment test criterion, examining sixteen neighboring pixels within a circular region around a candidate keypoint, denoted as *p*. If *n* neighborhood pixels are either brighter than $Ip + t$ or darker than $Ip - t$, where Ip represents the intensity of the candidate keypoint *p* and *t* signifies the threshold, *p* is classified as a corner. While this technique offers remarkable speed compared to alternative keypoint detection methods, it does exhibit sensitivity to noise, and orientation, and is reliant on the specific threshold value chosen.

Other methods for local feature detection include **Oriented FAST** and **Rotated BRIEF (ORB)**, which are faster in comparison to SURF and SIFT; however, it has constraints in terms of their descriptive capabilities and scale invariance under certain circumstances (Rublee, Rabaud, Konolige, & Bradski, 2011).

- **Maximally Stable Extremal Regions (MSER)** (Donoser & Bischof, 2006; Matas, Chum, Urban, & Pajdla, 2004) represent a collection of distinctive regions identified in a grayscale image. Each of these regions is characterized by an extreme property related to the intensity function in the region and along its outer boundary. MSER possesses specific qualities that contribute to it excellent performance as stable local detectors. The MSER set remains consistent under continuous geometric transformations and remains unaffected by affine changes in intensity. Moreover, MSERs are detectable at various scales.

- **Binary Robust Independent Elementary Features (BRIEF)** is a binary feature descriptor when the image block descriptor is generated by computing a simple intensity difference between pixels within an image block. The BRIEF keypoint feature descriptor is well-suited for real-time applications thanks to its rapid processing. Nevertheless, it exhibits limited flexibility to transformations like rotation, scaling, and image distortions (Calonder, Lepetit, Strecha, & Fua, 2010).

1.3.2. Mid-level features-based methods

Methods based on mid-level features are a type of feature-based approach that targets areas in an image exhibiting distinctive properties. These methods focus on detecting and utilizing features that are of a medium level of complexity, such as planes or blobs, which can be observed in the environment (Azzam, Taha, Huang, & Zweiri, 2020). The most critical characteristic of a feature is its repeatability, which allows it to be detected repeatedly when appearing in multiple frames taken from different viewpoints (Azzam et al., 2020).

Mid-level features hold an intermediate position in the visual feature hierarchy, being more complex than low-features such as contrast and spatial frequency, yet less complex than high-level features that encompass identifiable object components or complete objects (B. Long, Konkle, Cohen, & Alvarez, 2016). These characteristics possess the capacity to convey information about general category affiliation (B. Long et al., 2016). They have been used in various applications, such as image classification, word image representation, and simultaneous localization and mapping (SLAM) (Aslam et al., 2019; Azzam et al., 2020; Gordo, 2015).

These techniques present benefits in settings lacking distinct textures, which make it difficult to discern low-level features, as seen in scenarios like corridors. Nevertheless, there exists a trade-off between the precision of estimations and the duration required to generate precise models from the surroundings when employing mid-level feature-based methods (Azzam et al., 2020).

Here are popular examples of mid-level feature approaches:

a. Bag of Features (BoF)

The Bag of Features (BoF) (Often referred to as Bag-of-Visual-Words) is a method that represents images as orderless sets of local features. The name Bag of Features (BoF) is inspired by the Bag of Words representation used in textual information retrieval (O'Hara & Draper, 2011). This model stands as a widely adopted mid-level feature-driven approach in the fields of computer vision and image processing. It serves as a proficient method for representing images when it comes to image classification, image retrieval, object detection, and visual localization for robots (Hiba, Hamid, Jadida, & ALHEYANE Omar, 2016).

The Bag of Features model operates through the extraction of specific features from an image, followed by the quantization or clustering of these features. This is then used to construct a vocabulary comprising visual words, which are utilized to express the image as a sparse vector based on occurrence frequencies.

b. Spatial pyramids

Spatial pyramids represent a mid-level feature-based approach used in image recognition and object detection. They involve constructing orderless feature histograms over cells defined by a multi-level recursive image decomposition (Lazebnik, Schmid, & Ponce, 2006). Initially, the decomposition consists of just one cell (level 0), akin to a standard bag of feature representation. Subsequently, at level 1, the image is divided into four quadrants, generating four feature histograms and so forth (Lazebnik, Schmid, & Ponce, 2009). This method aims to provide an overarching framework that replaces ad-hoc implementation choices while remaining independent of specific features as long as they can be quantized to a discrete vocabulary. The spatial decomposition is tailored towards achieving approximate geometric matching goals (Lazebnik et al., 2009), making spatial pyramids a robust technique for addressing challenges in image recognition and object detection tasks.

Spatial pyramids have been utilized in a range of applications, including image classification, word image representation, and simultaneous localization and mapping (Lazebnik et al., 2006, 2009). The "spatial pyramid" that is generated serves as an uncomplicated and computationally effective expansion of an orderless bag-of-features image representation. It has demonstrated enhanced effectiveness in the recognition of natural scene categories (Lazebnik et al., 2006).

c. Upper units of convolutional networks or deep belief networks

The upper units in convolutional networks and deep belief networks play an important role in extracting complex, interpretable features that bridge the gap between low-level pixel information and high-level semantic understanding (Boureau, Bach, LeCun, & Ponce, 2010). These mid-level features are useful for tasks such as object recognition, image retrieval, and scene understanding. They provide a more abstract representation of the input, facilitate the interpretation and analyze the content of the images.

1.3.3. High-level features-based methods

High-level features are representations of information about the content of an image or that extend beyond basic visual characteristics, containing more complex and abstract information about the content of an image. These features capture interpretative responses, spatial, symbolic, semantic, or emotional characteristics to bridge the gap between low-level visual features and the semantics of the image.

High-level features in image classification and retrieval pertain to the visual attributes or features of an image that convey information about its content beyond basic low-level features like color and texture (Selvapattu & Nair, 2022). These features can be represented as feature vectors composed of numerical values. These numerical values are extracted using various techniques like deep learning, which involves neural networks and pattern recognition methods (Al-obaide & Al-ani, 2022). They offer a framework for image search, allowing images to be sorted based on visual appearance and similarity (Ciocca, Cusano, Santini, & Schettini, 2013). Content-based image retrieval uses high-level features to enhance the accuracy and effectiveness of image search by focusing on overall content and meaning (Tannaz & Sedghi, 2018).

Recently, deep learning is an advanced approach that has shown impressive results by deriving powerful features in various computer vision tasks. It obtains high-level (advanced) characteristics from entire image components, allowing for the direct learning of

detailed features through different techniques, which demonstrate the neural networks capability to acquire characteristic representations and patterns from visual contents, enabling them to carry out complex functions such as recognizing objects and classifying images.

- Deep learning models composed of multiple layers allow for the extraction of complex patterns and features from raw data, making them a promising approach for solving challenging problems and applied in various fields, especially image classification. Deep learning includes a variety of models, each designed to solve different types of problems such as Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), and Recursive neural networks (RvNNs). Later in the next chapter, these methods will be explored in detail.

1.4. Classification

The classification is a crucial step in different data analysis and machine learning techniques. It is used for organizing data and making accurate predictions in various fields such as image recognition, natural language processing, and predictive modeling. Thus, it helps to assign labels or categories to data for organizing and understanding the data by categorizing it into different groups or classes based on specific criteria or patterns.

The classification module consists of using a classification method to determine which class a data entry belongs to based on the descriptors extracted from the previous phase. It contains two phases: learning and testing. The former involves initializing the model base with descriptors and corresponding classes, while the latter entails assigning a class for each new example given whose class is unknown. These techniques are distinguished by their supervised or unsupervised nature depending on available information about "descriptor" observations. Supervised classification occurs when different classes of observations are known beforehand, which is preferred for image classification. On the other hand, if there is no prior knowledge about observation belonging to classes, it is referred to as unsupervised classification or clustering.

There are various algorithms commonly used for classification tasks, each with its specific strengths and weaknesses. The common examples include:

- Logistic Regression,
- Support Vector Machines (SVMs),

- Naive Bayes,
- Decision Trees,
- Random Forests,
- K-Nearest Neighbors (KNN),
- Neural Networks,
- Convolutional Neural Networks (CNNs),
- Recurrent Neural Networks (RNNs), and
- Transformers.

1.4.1. Logistic Regression

Logistic regression, also referred to as logit regression or the logit model, is utilized for forecasting the probability of categorical-dependent variables. It encompasses various forms such as binary logistic regression, multinomial logistic regression, and ordinal logistic regression (Verma, Charan, Fernando, & Ganesan, 2022). It is the most appropriate algorithm for analyzing the relationship between single or multiple predictors when data requires multiple classifications (Taherdoost, 2022). This algorithm is the best choice for a binary dependent variable because it works based on the occurrence chance. It determines the likelihood of a certain class or event by processing data using 0 or 1 values.

1.4.2. Support Vector Machine (SVM)

The support vector machine (SVM) (James, Witten, Hastie, & Tibshirani, 2013) is a robust machine learning algorithm capable of handling both multi-class classification and regression tasks. It is a classification approach that originated in the computer science field during the 1990s and has gained increasing acceptance. SVMs have demonstrated strong performance across diverse scenarios and are frequently regarded as one of the top pre-built classifiers. Furthermore, the support vector machine is an extension of a basic and straightforward classifier known as the maximal margin classifier. While this classifier is elegant and uncomplicated, it observes that cannot be utilized for most datasets, as it necessitates linear boundary to separate the classes (James et al., 2013).

In cases where the data can be separated linearly, linear regression offers an effective algorithm for separating the data using a hyperplane. However, there are often multiple hyperplanes capable of separating the data, and in this regard, SVM selects the most optimal one. The chosen hyperplane by SVM has the largest margin from the data points on both sides. Another benefit is that this algorithm only necessitates knowledge of support vectors

or points located on the margin (Ashfaque & Iqbal, 2019). Suppose the training samples are provided by

$$D = \{(x_1, y_1), (x_2, y_2), \dots, (x_N, y_N)\} = D_+ \cup D_- \quad (29)$$

Points in D_+ have last coordinate +1, while points in D_- have last coordinate -1. We define ε as the subset of X such that

$$E = \{x_1, x_2, \dots, x_N\} = E_+ \cup E_- \quad (30)$$

$x \in E_+$ if and only if $f(x) = +1$.

The hypothesis function approximating f is

$$h_w(x) = w^T x + c \quad (31)$$

Such that

$$h_w(x) = \begin{cases} > 0 & \text{for} \quad x \in D_+ \\ < 0 & \text{for} \quad x \in D_- \end{cases} \quad (32)$$

1.4.3. Naive Bayes

Naive Bayes classifiers (Rish, 2001) are linear models known for their simplicity and high efficiency. These classifiers are based on Bayes' theorem and make the naive assumption that features in a dataset are mutually independent. In practical applications, the assumption of independence is often violated, but naive Bayes classifiers still demonstrate strong performance despite this unrealistic assumption. Naive Bayes classifiers have the potential to outperform more sophisticated alternatives, particularly when dealing with small sample sizes (Domingos & Pazzani, 1997).

The naive Bayes assumption (Murphy, 2006) posits that each feature is conditionally independent given the class label:

$$p(x|y = c) = \prod_{i=1}^{D} p(x_i|y = c) \quad (33)$$

Even though this is typically not the case (due to the usual dependence of features), the resulting model is straightforward to fit and performs remarkably well. For Gaussian data, we obtain

$$p(x|y = c, \theta_c) = \prod_{i=1}^{D} N(x_i|\mu_{ic}, \sigma_{ic}) \quad (34)$$

So it simply needs to calculate C × D individual Gaussian parameters, μ_{ic} and σ_{ic}. For binary data, this results in

$$p(x|y = c, \theta_c) = \prod_{i=1}^{D} Be(x_i|\theta_{ic}) \quad (35)$$

Finally, it just needs to estimate separate Bernoulli parameters, θ_{ic}, for C × D.

1.4.4. Decision Trees

Decision trees (James et al., 2013) can be utilized for both regression and classification tasks. It is a powerful and popular classification algorithm. A classification tree resembles a regression tree, but it is utilized for predicting a qualitative outcome instead of a quantitative one. While a regression tree predicts the response by averaging the responses of training data in the same leaf node, in contrast, a classification tree assigns each observation to the most frequent class among training observations within its region. When analyzing outcomes from a decision tree, our focus is not just on forecasting categories for individual end points, but also on comprehending the assortment of various groups within those areas according to the training data.

Decision trees aim to identify the most informative descriptive features related to the target feature and subsequently divide the dataset based on these features' values to achieve as much purity as possible in the resulting underlying datasets. The descriptive characteristics that best mirror the target/output features are typically considered to be the most informative. The search for these informative characteristics persists until a certain criterion is met, resulting in the formation of what is commonly referred to as terminal nodes (leaf nodes). A typical decision tree consists of main root node, interior nodes, as well as terminal leaf nodes. These components are linked by branches or edges. The predictions for new input examples are kept in the terminal leaf nodes, as our model has developed an understanding of the underlying patterns in the training data. This allows it to make inferences and forecast the target feature value (class) for unseen input examples.

1.4.5. Random Forests

Random forests (Breiman, 2001) represent a robust ensemble learning method rooted in decision trees, leveraging the collective predictions of multiple trees to enhance accuracy and resilience beyond that of individual trees. Conceptually, it can be likened to a panel of diverse experts casting their votes on a prediction, with the majority consensus being decisive.

1.4.6. K-Nearest Neighbors (KNN)

The *k*-Nearest Neighbors (*k*NN) is a non-parametric and supervised learning classifier, known for its simplicity and effectiveness in numerous cases. When classifying a data record *t* using the *k*-nearest neighbors method, a neighborhood of *t* is formed by retrieving its *k* nearest neighbors. Typically, classification for *t* is determined through majority voting among the data records in the neighborhood, with or without consideration of distance-based weighting. The choice of an appropriate value for *k* is crucial when applying the *k*NN method, as the success of classification depends heavily on this value. In a way, the *k*NN method's outcome is influenced by *k*. There are various methods to select the value for *k*; one simple approach involves running the algorithm multiple times with different values of *k* and selecting the one that yields optimal performance (Guo, Wang, Bell, Bi, & Greer, 2003).

KNN is a simple and easy-to-understand algorithm with several strengths. These include ease of implementation, flexibility, and adaptability. But, it also has some weaknesses such as high memory and computational requirements, sensitivity to irrelevant or redundant features, and the need for careful selection of *k* and distance metrics.

1.4.7. Neural Networks (NNs)

Neural networks (Cocco, Monasson, & Zamponi, 2022) are a form of machine learning model inspired by the construction and function of the human brain. They comprise interconnected nodes known as neurons, arranged in layers. Each neuron receives input, conducts a computation, and generates an output. These outputs from one layer are then used as inputs for the next layer, facilitating the network's ability to grasp intricate patterns and relationships within the data.

Neural networks are employed in supervised learning assignments, where the network is trained to associate input data with output labels using a collection of training samples. The network modifies the weights and biases of its neurons through an iterative

process known as backpropagation, which involves continuously adjusting the parameters to reduce the disparity between predicted outputs and actual outputs. This enables the network to generalize and produce precise forecasts for unseen data.

1.4.8. Convolutional Neural Networks (CNNs)

Convolutional Neural Networks (CNNs) are a specific type of neural network architecture (NNs), known as deep neural networks, and they are particularly well-suited and effective for image classification tasks. CNNs can extract features from raw pixel data of images and learn to classify them. The basic architecture of a CNN consists of convolutional layers, ReLU activation function, pooling layers, and fully connected layers for classification. Their ability to extract features at various levels of abstraction and perform convolution operations using a hierarchical architecture of layers accounts for their success in capturing spatial features and patterns in images. When employing a CNN for image classification, it is essential to start with training the model on a huge dataset comprising labeled images that feature the objects of focus. Throughout this training phase, the CNN learns to associate the extracted features with their corresponding labels using back propagation and optimization. Subsequently, following its training, the CNN becomes capable of generating predictions for new, unseen images by running them through the network and selecting the label associated with the highest predicted probability (Krichen, 2023).

1.4.9. Recurrent Neural Networks (RNNs)

Recurrent Neural Networks represent a sort of neural network which incorporates recurrence, enabling them to leverage data from previous iterations through the network. Designed for processing sequential data such as speech or text, they have been utilized in tasks like speech recognition and language translation. However, their performance is generally inferior to that of CNNs when it comes to visual data analysis. This difference arises from RNNs' limited capacity to capture the spatial relationships between pixels in an image, a crucial feature of CNNs. Moreover, RNNs tend to be more computationally intensive than CNNs, particularly when dealing with lengthy sequences, rendering them less feasible for certain applications. RNNs are frequently employed in a variety of applications and exist in diverse forms such as Long Short-Term Memory (LSTM) and Gated Recurrent Units (GRUs) (Krichen, 2023).

1.4.10. Transformers

Transformers are a kind of Deep Neural Network (DNN) that employ self-attention to comprehend contextual relationships in sequential data. In contrast to conventional neural networks and different forms of RNNs, such as LSTM, Transformer models are adept at handling lengthy dependencies between input sequence elements and enabling parallel processing. They have been utilized in tasks like Natural Language Processing (NLP), computer vision (CV), Multi-modal applications, Audio and Speech processing, and Signal Processing (S. Islam et al., 2024).

1.5. Difficulties in computer vision

There are numerous reasons why computer vision is quite complex; we cite (Sonka et al., 2015):

1.5.1. Loss of information in 3D → 2D

The loss of information following the transition from 3D to 2D, in fact, is a phenomenon that occurs during image capture by a camera. The geometric properties of the human eye and the camera have been approximated for centuries by a pinhole model consisting of a box with a small hole. One problem with the pinhole model is that the projective transformation sees a small object close to the camera in the same way as a large object at a distance from the camera. In this case, a human being uses a standard to guess the real size of the object, but the computer does not.

1.5.2. Interpretation of image(s)

The interpretation of images is the main tool of computer vision, tackling problems that humans solve unknowingly. When a human being interprets an image, he or she brings previous knowledge and experience to bear on current observation. Indeed, human beings can represent knowledge over the long term and use it to solve new problems. Artificial intelligence has invested several decades in attempts to equip computers with the ability to understand observations, and while progress has been enormous, the practical ability of a machine to understand observations remains very limited. From a mathematical logic and linguistic point of view, image interpretation can be seen as cartographic interpretation:

$$\textit{interpretation: image data} \rightarrow \textit{model}$$

The (logical) model means a specific world in which the observed objects make sense. An example might be cell nuclei in a biological sample, rivers in a satellite image, or parts in any industrial process being checked for quality. There may be several interpretations of the same image. The integration of interpretation with computer vision enables us to use the concepts of mathematical logic, linguistics as well as syntax and semantics.

1.5.3. Noise

The noise is an inherent part of every measurement in the real world. It calls for mathematical tools that can eliminate it. On the other hand, more complex tools make image analysis much more complicated than standard methods.

1.5.4. Too much data

Images and videos have become major focuses of vision applications due to their large file sizes. However, advancements in technology have reduced the impact on processor and memory requirements, making consumer-level products adequate for many tasks. Despite this progress, achieving real-time performance remains a challenge for many applications.

1.5.5. Brightness measured

The measurement of brightness in images is determined by complex physics related to the formation of the image. The radiance, which is approximately equivalent to brightness or image intensity, relies on factors such as light source type and intensity, as well as its position. Additionally, it is influenced by the observer's position, surface local geometry, and reflectance properties of the surface.

1.5.6. Local window vs. need for the global view

Image analysis algorithms commonly examine a specific storage bin within operational memory, for example, the pixel in the image, and its surrounding area. This limited perspective can make it challenging to comprehend the overall context of an image. Interpreting an image solely based on local aspects or with access to only a few localized perspectives is often very complex.

1.6. Conclusion

We have presented, throughout this chapter, basic concepts of image analysis and computer vision. In addition, we explored the phases of feature extraction and classification and their known approaches studied in the literature for each phase. We have explained feature extraction methods based on low-level features, mid-level features, and high-level features. Also, we have discussed the classification methods. Finally, we have cited some reasons that make computer vision difficult.

In the next chapter, we will discuss the basics of artificial intelligence, machine learning and deep learning and the relation between them.

CHAPTER 2

MACHINE LEARNING AND DEEP LEARNING

CHAPTER 2: MACHINE LEARNING AND DEEP LEARNING

2.1. Introduction

Artificial intelligence aims to make machines intelligent by simulating human problem-solving capabilities for tasks with high logical or algorithmic complexity. Machine learning, a major branch of artificial intelligence, enables machines to learn decision-making from observations and use data analysis. In machine learning, programs analyze sets of data to create models for drawing conclusions about new data. Deep learning is a subset of machine learning that employs neural networks to solve problems in a hierarchical manner by identifying the most significant features.

In this chapter, we will first present definitions and basic concepts of artificial intelligence, machine learning and deep learning. Then, we explained the relation between machine learning and deep learning. Subsequently, we review some types of deep learning networks, the CNN structure, and the most frequent methods for object classification using deep learning. Finally, we cited common CNN model architectures.

2.2. Artificial intelligence

The domain of artificial intelligence concentrates on empowering computers to execute tasks that were traditionally only carried out by humans, using advanced techniques (Caiming Zhang & Lu, 2021). In recent years, people's lifestyles have changed as a result of artificial intelligence's rapid development (P. Wang, 2019). Artificial intelligence (AI) has emerged as a significant strategy for global nations, bolstering their competitiveness and safeguarding national security (Caiming Zhang & Lu, 2021). Numerous nations have implemented favorable measures and enhanced the allocation of essential technologies and skilled personnel to gain a competitive edge in a fresh wave of global rivalry (Haenlein & Kaplan, 2019). It has become a focal point of research in the field of science and technology (Caiming Zhang & Lu, 2021).

Artificial intelligence is an interdisciplinary field that can incorporate cognitive processes, machine learning, emotional understanding, human-computer interaction, storage of data and decision-making (Yan, 2016). John McCarthy first introduced the concept at the Dartmouth Conference in the middle of the 20th century. Since 1993, significant advancements have been made in the field of artificial intelligence. The rapid development

of the neural network is attributed to the extensive utilization of the back propagation algorithm. This widespread application of expert systems in large-scale environments has resulted in significant cost savings and enhanced industry efficiency (B. Huang, Huan, Xu, Zheng, & Zou, 2019).

The field of artificial intelligence has undergone an extensive period of advancement, spanning over 70 years. Its progression can be categorized into distinct phases: the inception of the artificial neuron model in 1943 marked the beginning of research into artificial neural networks. The birth of artificial intelligence can be traced back to the Dartmouth Conference in 1956 when the concept was first introduced. This event marked a significant increase in research on artificial intelligence by the global academic community and led to frequent academic exchanges during this period. During the 1960s, there was a decline in the popularity of connectionism and subservience, leading to a setback in the development of advanced technology. The exploration of the back propagation algorithm commenced in the 1970s while increasing costs and computing capabilities posed challenges for research and implementation of expert systems. Advancing further posed challenges, yet there was a gradual emergence of progress in artificial intelligence. During the 1980s, back propagation neural networks gained significant recognition, leading to rapid advancement in algorithm research based on artificial neural networks. Simultaneously, there were substantial improvements in computer hardware functionality and the expansion of the Internet facilitated the development of artificial intelligence. The emergence of mobile Internet in the early 2000s led to an increase in artificial intelligence application possibilities. A major milestone occurred in 2012 with the introduction of deep learning, which significantly advanced the development of artificial intelligence. This new algorithm achieved significant progress in speech and visual recognition capabilities (Caiming Zhang & Lu, 2021).

2.3. Machine learning

The fundamental concept of machine learning involves utilizing an algorithm that enhances its effectiveness by acquiring knowledge from data (Salim Malek, 2018). The key problem areas addressed through machine learning include clustering, prediction, dimensionality reduction, and classification (Erhan, Courville, Bengio, & Vincent, 2010). In terms of categorization approaches learning, machine learning could be segmented into four groups: supervised, unsupervised, semi-supervised, and reinforcement learning (Bose, 2017).

Supervised learning involves utilizing labeled data for training to make predictions about the type or value of new data. Based on the different types of predictions made, it can be categorized into classification and regression. Common methods used in supervised learning include Support Vector Machines and linear discrimination techniques. Regression problems involve predicting continuous output values, such as analyzing housing price data to fit a curve based on sample input and making predictions. On the other hand, classification problems entail forecasting discrete output values, like determining if a photo features a dog or a cat based on specific characteristics, resulting in an output value of 1 or 0 (Najafabadi et al., 2015; Neyshabur, Bhojanapalli, McAllester, & Srebro, 2017).

When data lacks labels, unsupervised learning is utilized for data mining. Unsupervised learning focuses mainly on clustering, where data can be grouped based on different attributes without any specific tags. Common methods of unsupervised learning include k-clustering and principal component analysis. K-clustering relies on the ability to measure differences between data using Euclidean distance; if this is not feasible, the data needs to be transformed into a format that allows for such measurement. On the other hand, principal component analysis is a statistical method that involves transforming correlated variables into uncorrelated ones through orthogonal transformation; these newly derived variables are known as principal components. The fundamental concept behind this approach is to substitute the originally related indicators with a set of independent comprehensive ones (Caiming Zhang & Lu, 2021).

Semi-supervised learning can be understood as a combination of supervised and unsupervised learning, where labeled and unlabeled data are utilized in the training process. Typically, there is a larger amount of unlabeled data compared to labeled data. Although the concept of semi-supervised learning is promising, it is not widely applied in practical scenarios. The commonly used algorithms for semi-supervised learning consist of self-training, graph-based approaches, and semi-supervised support vector machines (S3VM) (Caiming Zhang & Lu, 2021).

Reinforcement learning involves obtaining rewards through interacting with the environment, evaluating actions based on their reward outcomes, and then training the model. The significance of exploration and exploitation in reinforcement learning poses a complex problem: to maximize rewards, individuals need to select actions likely to yield the highest reward while also discovering unknown actions (Caiming Zhang & Lu, 2021).

2.4. Deep learning

Deep learning, a subset of machine learning, involves interpreting data and combining multiple layers of features to generate predictive results. It has demonstrated strong performance across various industries, particularly in the areas of image classification, object recognition, and segmentation. Deep learning algorithms have remarkably improved the effectiveness of precise classification tasks that aim to differentiate between subcategories (S. Sharma & Guleria, 2022). It employs a large, complex system of artificial neural networks that perform computations using continuous representations, similar to the hierarchically structured neurons found in human brains. This approach is currently the most effective method for machine learning, applicable to all kinds of machine learning tasks. It demonstrates improved generalization capabilities with limited data and better scalability when handling large datasets and computational resources.

Deep learning places a strong emphasis on constructing models with multiple hidden layers to emphasize the significance of feature extraction; this latter is a crucial phase in pattern-matching systems, particularly in image classification. The accuracy of feature extraction directly influences the recognition rate. Deep learning effectively captures characteristics through iterative feature translation, automatically extracting features from both text and images. It offers the benefit of processing large amounts of information and has experienced significant growth in recent years. Deep learning has been successfully utilized across a wide range of applications including NLP, cybersecurity, text mining, image classification, video recommendation, robotics, and bioinformatics. Notably surpassing various traditional machine learning models in numerous sectors, deep learning (DL) is also known as Representation Learning (RL) (S. Sharma & Guleria, 2022).

2.4.1. Relationship between machine learning and deep learning

Artificial intelligence seeks to impart intelligence to machines by imitating human problem-solving abilities for tasks requiring high logical or algorithmic complexity. Machine learning, a key aspect of artificial intelligence, allows machines to acquire decision-making skills from observations and data analysis. Within machine learning, algorithms analyze datasets in order to construct models for making inferences about new data. Deep learning is a part of machine learning that utilizes neural networks to address problems hierarchically by recognizing the most important features as illustrated in **Figure 2.1**.

The performance of machine learning algorithms relies heavily on the input- data representation quality. Studies have demonstrated that the well-crafted data representation

leads to better performance as compared to an inadequate one. As a result, feature engineering has been a major focus of research in machine learning for many years, with numerous studies exploring this approach. Feature engineering involves creating features from raw data and is highly specific to each field, often demanding substantial human effort (Alzubaidi et al., 2021).

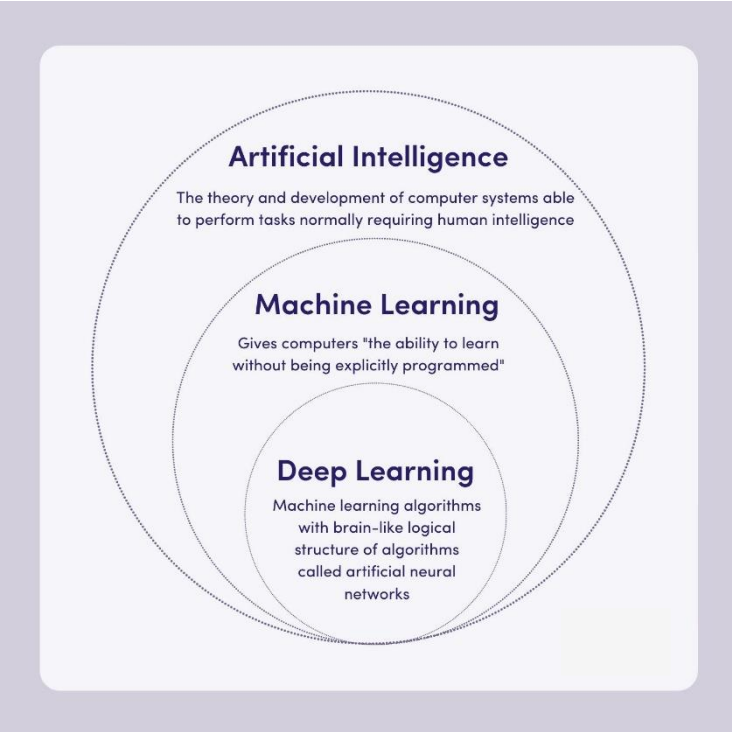

Figure 2.1. Relation between Artificial intelligence, machine learning and deep learning

Deep learning algorithms are designed with a multiple-layer data representation structure, where initial layers capture the basic features, which are low-level features, and later layers focus on more complex ones, which are high-level features. This architectural concept draws inspiration from artificial intelligence and mimics the way sensory regions in the human brain process information. Just like humans automatically perceive data from various scenes, deep learning models utilize input scene information to generate classified objects as output. In summary, this approach replicates the cognitive processes of the human brain, highlighting one of the key advantages of deep learning (Alzubaidi et al., 2021). The **Figure 2.2.** illustrates the difference between machine learning and deep learning algorithms.

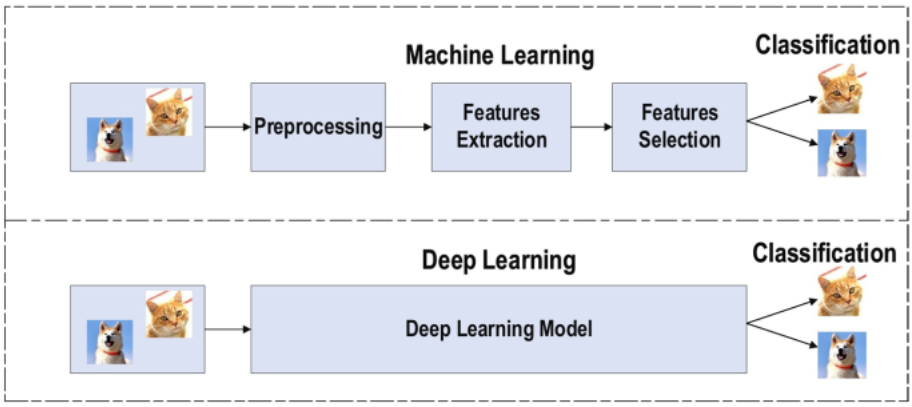

Figure 2.2. The difference between machine learning and deep learning (Alzubaidi et al., 2021)

2.4.2. Types of deep learning networks

This part covers the deep learning networks popular types, which encompass recurrent neural networks (RNNs), recursive neural networks (RvNNs), as well as Convolutional neural networks (CNNs). While this section provided a brief explanation of RvNNs and RNNs, it deeply explained CNNs due to their significant importance in various applications as one of the most widely used network types.

- **Recursive neural networks (RvNNs)**

RvNNs are capable of making predictions in a hierarchical arrangement and categorizing the results using compositional vectors. They are designed to process objects with irregular structures for example graphs or trees, producing a consistent distributed representations from variable- size recursive data structures. The network undergoes training utilizing a back-propagation by structure (BTS) learning system. The BTS system employs a similar approach to the general back-propagation algorithm and is capable of accommodating a tree-like structure. Auto-association trains the network to renew the pattern of the input-layer at the output layer. RvNN demonstrates high effectiveness in the applications of natural language processing (NLP) (Alzubaidi et al., 2021). An example of a RvNN tree is shown below in Figure 2.3.

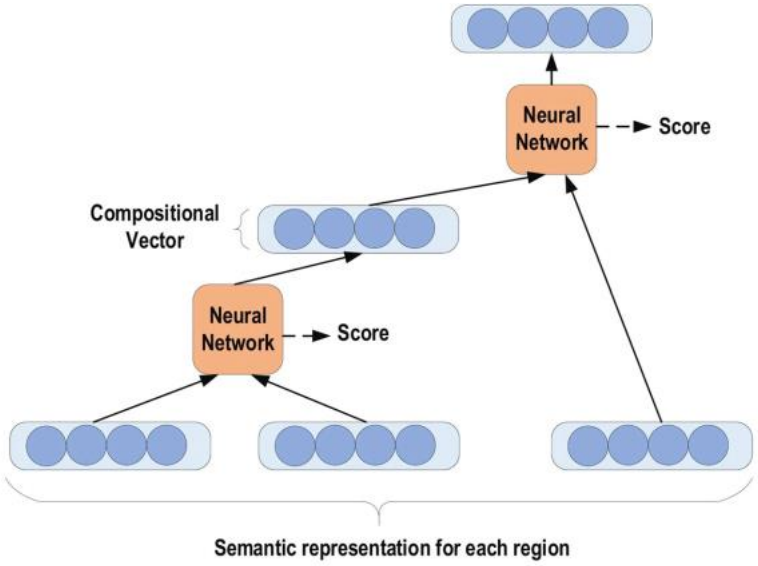

Figure 2.3. Example of RvNN tree

- **Recurrent neural networks (RNNs)**

RNNs are a widely used and well-known algorithm in the field of DL. RNN is commonly utilized in speech processing and NLP cases, leveraging consecutive data within the network. The established structure within the series provide valuable information crucial to various applications such as understanding sentence context for word meaning determination. In this view, RNN can be likened to a form of short- term memory unit with x denoting the input layer, y representing the output layer, and s indicating the state (hidden) layer (Alzubaidi et al., 2021). A typical unfolded diagram of an RNN for a given input sequence is depicted in Figure 2.4.

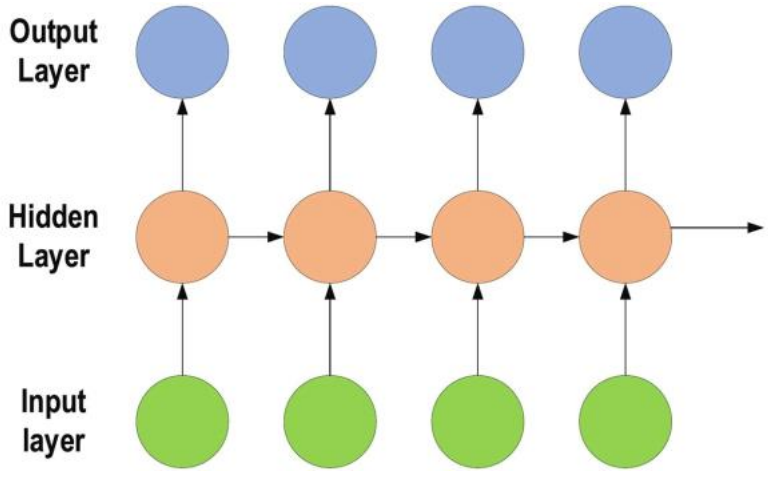

Figure 2.4. Diagram of typical unfolded RNN

- **Convolutional neural networks (CNNs)**

CNNs are the most well-known and frequently used algorithms. They offer a significant advantage over earlier methods by automatically recognizing important features without human intervention. CNNs have been widely used across various fields such as computer vision, location recognition, speech processing, and facial recognition (Krichen, 2023).

The design of CNNs draws inspiration from the neurons in human and animal brains, resembling that of a conventional neural network (Alzubaidi et al., 2021). An instance of CNN structure for classification of images is depicted in Figure 2.5.

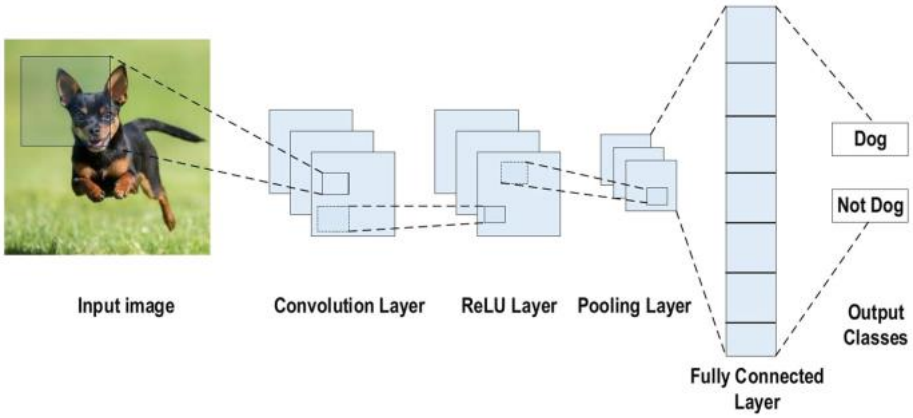

Figure 2.5. CNN architecture for image classification

The input *x* of each layer in a CNN model is structured in three dimensions: height, width, and depth, represented as $m \times m \times r$. The height *(m)* is equal to the width while the depth corresponds to the channel number. For instance, in an RGB image, the depth *(r)* is three. Multiple kernels (filters) present in each convolutional layer are denoted by *k* and have three dimensions *($n \times n \times q$)*, similar to the input image; however, with *n* being smaller than *m* and *q* either equal to or less than *r*. These kernels form local connections that share bias b^k and weight W^k parameters for generating *k* feature maps h^k with a size of *(m-n-1)* each when convolved with input data. The dot product between its input and weights is computed by the convolution layer as expressed in Eq. 1. Following, by leveraging nonlinear transformations or activation functions to generate the corresponding outputs:

$$h^k = f(W^k \times x + b^k) \qquad (2.1)$$

The subsequent stage involves reducing the resolution of each feature map in the sub-sampling layers. This results in a decrease in the network parameters, expediting the training process and consequently addressing overfitting. A pooling function (e.g., max or average) is then applied to neighboring areas of size p×p for all feature maps, when p denotes kernel size. Subsequently, the fully connected layers receive mid- level features and low- level features to create high-level abstractions, representing the final-stage layers similar to those found in a standard neural network. The classification scores are produced by using an ending layer [e.g., support vector machines or softmax]. Each score indicates the probability of a specific class for a given instance.

2.5. CNN architecture

The Convolutional Neural Network is created to handle visual data like images and videos. Its architecture consists of several layers that serve different purposes, including the convolution layer, pooling layer, fully connected layer, and dropout layer. Additionally, it incorporates activation functions like sigmoid, ReLU, and SoftMax with distinct ranges. Each element in CNN structure and its role is characterized in particular below.

2.5.1. Convolutional Layer

The convolutional layer (Alzubaidi et al., 2021; Krichen, 2023) is considered the main element in CNN architecture, which contains a set of filters known as kernels. These filters are used to convolve the input image, represented as N-dimensional tensors, and produce the feature map of output.

- The kernel is defined as a grid of discrete numerical values, each known as kernel weight. The random numbers are initially allocated to serve as a weights during the start of CNN training. Various methods are also employed for initializing these weights. Subsequently, adjustments to these weights occur in each training epoch, enabling the kernel to learn and extract important features.
- The convolutional operation begins with describing the CNN input format, which differs from the vector format used in traditional neural networks. CNN takes a multi-channeled image as input, such as single-channel for gray-scale and three-channeled for RGB images. To illustrate, consider a 4×4 gray-scale image and a 2×2 randomly initialized kernel for understanding the convolutional operation. The kernel slides over the entire image both horizontally and vertically, computing dot products

between the input image and the kernel to create individual scalar values through multiplication and summation of corresponding elements. This iterative process continues until no additional sliding can be attained, generating feature map values as output. The Figure 2.6. illustrates the main computations performed at each stage of the convolutional layer.

- Sparse Connectivity: In fully connected neural networks, every neuron in a layer is connected to all neurons in the subsequent layer. In Contrast, convolutional neural networks have only limited connections between adjacent layers, resulting in fewer required weights and reduced memory storage needs. This approach proves efficient for memory usage due to the small number of connections. Moreover, compared to matrix operations prevalent in FC networks, CNNs benefit from less computationally intensive dot product operations.
- Weight Sharing: In a CNN, there is no specific assignment of weights between individual neurons in neighboring layers. Instead, the entire set of weights operates across all pixels of the input matrix. This approach reduces training time and costs by learning a single set of weights for the entire input, eliminating the need to learn additional weights for each neuron.

2.5.2. Pooling Layer

The primary function of the pooling layer is to down-sample the feature maps produced by convolutional operations, effectively reducing their size while preserving most of the essential information (Krichen, 2023). Similar to convolutional operations, pooling involves initially assigning a stride and kernel size for the operation. Various types of pooling methods such as tree pooling, gated pooling, average pooling, min pooling, max pooling, global average, and global max are available for use in different scenarios (Galanis, Vafiadis, Mirzaev, & Papakostas, 2022). The most commonly used techniques include max and min pooling along with global average pooling. Figure 2.7 demonstrates these three (3) sorts of pooling operations.

The pooling layer can lead to a decrease in the overall performance of the CNN, as it primarily focuses on identifying the correct location of a feature within an input image without adequately capturing all relevant information, which is a primary limitation of this layer.

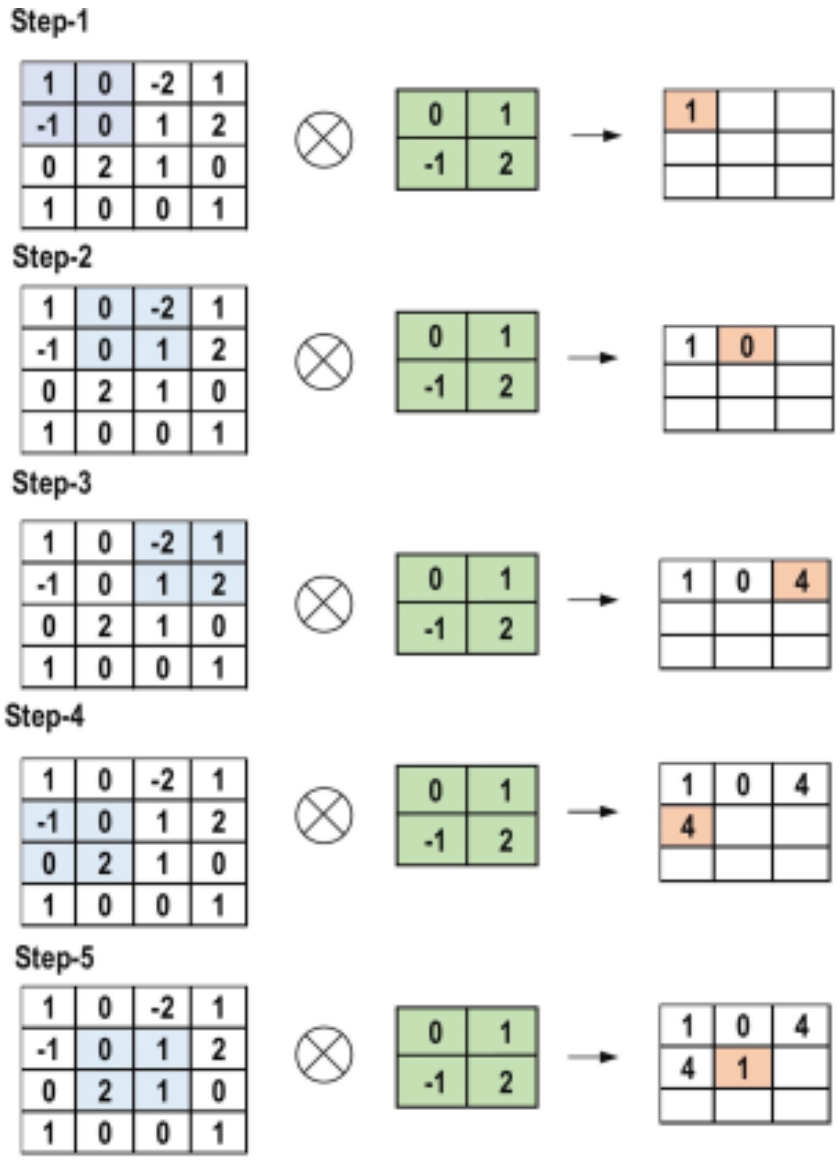

Figure 2.6. The main computations performed at each stage of the convolutional layer

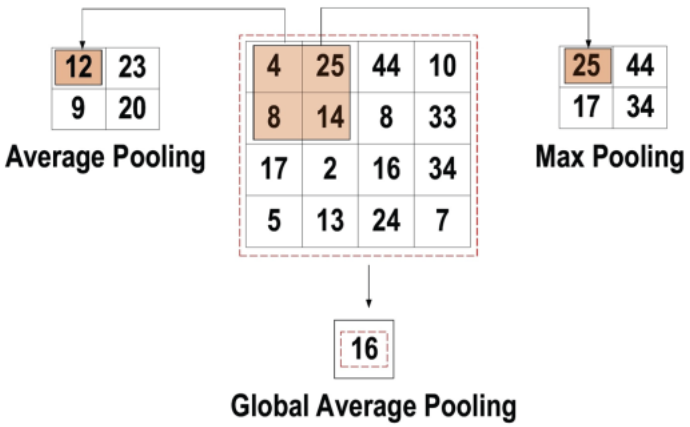

Figure 2.7. Types of pooling operations

2.5.3. Activation function

Activation functions (Alzubaidi et al., 2021) are essential in mapping input to output in all neural network types. If it is applicable and by calculating the neuron inputs weighted summation and their biases, the input value is determined. Consequently, activation functions decide whether a neuron should be activated based on a specific input, generating the correspondent output.

Activation functions non- linear are utilized after each layer with weights in the CNN architecture. These activation functions non-linear properties ensure that the transformation from input to output is non-linear, enabling the network to learn complex patterns. Additionally, the activation function needs to be differentiable, as this enables error back-propagation for network training. The most commonly employed types of activation functions in CNN and other deep neural networks include:

- **Sigmoid** activation function takes real numbers as input and limits the output to a range from zero to one. The curve of the sigmoid function resembles an S-shape and can be expressed mathematically by Eq. 2.2.

$$f(x)_{sigm} = \frac{1}{1+e^{-x}} \quad (2.2)$$

- **Tanh** function resembles the sigmoid function in that it takes real numbers as input, but its output is confined to a range between -1 and 1. Its mathematical expression is shown in Eq. 2.3.

$$f(x)_{tanh} = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}} \quad (2.3)$$

- **ReLU** is the most frequently utilized function in the context of Convolutional Neural Networks. It transforms all input values to positive numbers, providing a key advantage of lower computational workload compared to other functions. Its mathematical expression is shown in Eq. 2.4.

$$f(x)_{ReLU} = \max(0, x) \quad (2.4)$$

2.5.4. Fully Connected Layer

This layer is typically situated following each convolutional neural network (CNN) architecture (Yang, Yang, Hou, & Su, 2021). Within this layer, every neuron is linked to all neurons in the prior layer using the Fully Connected method. It serves as the classifier for the CNN and operates similarly to a standard multi- layer perceptron neural network, functioning like a kind of feed- forward ANN. The input for the Fully Connected layer is

derived from either the convolutional or last pooling layer, presented in vector form after flattening feature maps. The output from the FC layer signifies the ultimate CNN output, as depicted in Figure 2.8.

2.5.5. Loss function

The output layer—the last layer in the CNN architecture—is responsible for achieving the final classification. To determine the expected error generated throughout the training samples in the CNN model, certain loss functions are used in the output layer. This error indicates the difference between the actual and expected output. The CNN learning process will then be used to optimize it.

Nevertheless, the error is computed by the loss function using two parameters. The first parameter is the CNN estimated output, also known as the prediction. The second parameter is the actual output, also called the label. Different kinds of loss functions are used for various types of problems. A brief explanation of a few of the loss function types is provided below.

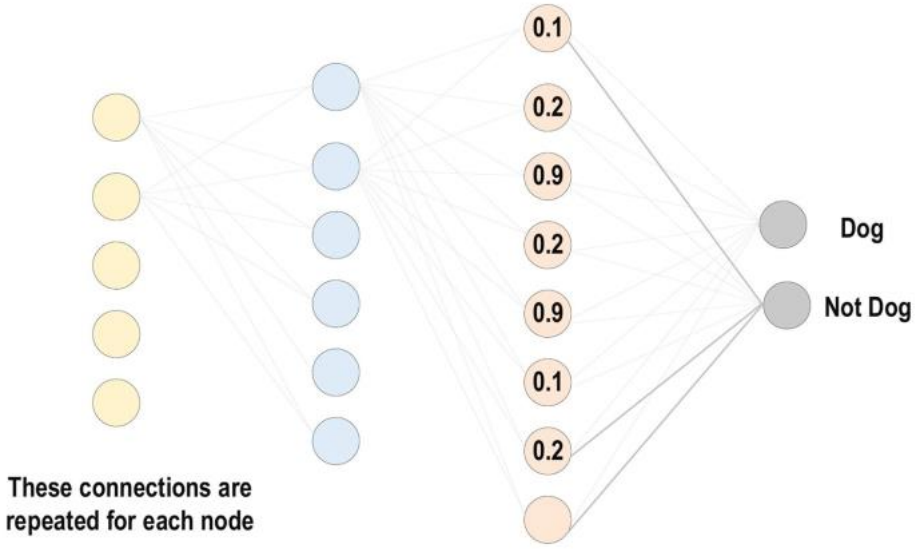

Figure 2.8. The fully connected layer (FC)

- **Cross-Entropy or Softmax Loss Function:** The CNN model loss function is typically measured using this function. The probability p ∈ {0, 1} is its output. Furthermore, it is typically used in multi-class classification problems as an alternative to the square error loss function. Softmax activations are used in the output layer to produce the output within a probability distribution. Eq. 2.5 is the mathematical expression for the output class probability:

$$p_i = \frac{e^{a_i}}{\sum_{k=1}^{N} e_k^a} \tag{2.5}$$

Here, N is the number of neurons in the output layer, and e^{a_i} is the non-normalized output from the layer before it. Lastly, Eq. 2.6 provides a mathematical representation of the cross-entropy loss function.

$$H(p, y) = -\sum_i y_i \log(p_i) \quad where \quad i \in [1, N] \tag{2.6}$$

- **Euclidean Loss Function:** In regression issues, this function is commonly utilized. Furthermore, it is also known as the mean square error. Equation 2.7 is the mathematical representation of the estimated Euclidean loss.

$$H(p, y) = \frac{1}{2N} \sum_{i=1}^{N} (p_i - y_i)^2 \tag{2.7}$$

- **Hinge Loss Function:** This function is frequently used to solve binary classification-related issues. This issue concerns maximum-margin-based classification; it is particularly significant for Support Vector Machines (SVMs) that employ the hinge loss function, in which the optimizer aims to maximize the margin surrounding two objective classes. Equation 2.8 is its mathematical formula.

$$H(p, y) = \sum_{i=1}^{N} \max(0, m - (2y_i - 1)p_i) \tag{2.8}$$

Typically, the margin m is configured to 1. Additionally, the desired output is represented by y_i, and the predicted output by p_i.

2.6. Classification using Deep Learning

Deep Learning removes the necessity for manual feature extraction, negating the requirement to identify features utilized in image classification. Instead, Deep learning methods extract features directly from input data such as images, videos, and sound. The pertinent features are not pre-trained; they are acquired during the network's training process with a set of images. This automated feature extraction contributes to making deep learning models very precise for computer vision assignments like object classification. Three of the most frequently utilized methods for object classification using deep learning include ("What Is Deep Learning?," n.d.):

2.6.1. Training from Scratch

Training a deep network from the beginning involves assembling an extensive labeled data set and developing a network structure that can effectively learn the features and model. This method is suitable for new applications or those with a significant number of output categories. However, this approach is less frequently used due to the lengthy training duration required as these networks typically take days or even weeks to train, given their substantial volume of data and learning rate.

2.6.2. Transfer Learning

Most deep learning applications commonly adopt the transfer learning technique, which entails adapting a pre-existing model. This process begins with utilizing an established network like AlexNet or GoogLeNet and inputting new data that includes classes not previously encountered. By making adjustments to the network, it becomes possible to undertake a new task, for instance, classifying only dogs or cats rather than 1000 different objects. Additionally, this method requires significantly less data (processing thousands of images as opposed to millions), resulting in reduced computation time ranging from minutes to hours.

2.6.3. Feature Extraction

A less frequently used and more specialized method for deep learning involves utilizing the network as a tool for extracting features. As each layer is responsible for learning specific image attributes, these features can be extracted from the network at any point during training. Subsequently, these extracted features can serve as input for a machine learning model like support vector machines.

2.7. CNN models architectures

Over the past decade, there have been numerous advancements in CNN architectures. The design of the model plays a crucial role in enhancing performance across various applications. Changes to CNN architecture since 1989 have involved structural restructuring, regularization techniques, parameter optimizations, and more. Notably, significant improvements in CNN performance can be attributed to reorganizing processing units and introducing innovative blocks. In particular, innovations in network depth represent some of the most notable developments in CNN architectures.

2.7.1. Alexnet

Krizhevesky et al. (Krizhevsky, Sutskever, & Hinton, 2012) initially introduced AlexNet, which led to enhancements in the learning capacity of CNN by elevating its depth and incorporating various parameter optimization techniques. This deep CNN structure has garnered significant admiration for achieving innovative outcomes in image classification and recognition. **Figure 2.9** presents the fundamental framework of AlexNet. The capability learning of deep Convolutional Neural Network was constrained by hardware limitations during this period. To address these restrictions, two NVIDIA GTX 580 GPUs were employed simultaneously in trainning AlexNet. Additionally, to broaden the CNN applicability across various image classes, the feature extraction stages number was raised from five (5) in LeNet to seven (7) in AlexNet. Although increased depth contributes to improved generalization for multiple image resolutions, overfitting emerged as a primary challenge associated with depth expansion. Krizhevesky et al.'s method includes randomly passing over various transformational units during the training stage to ensure that the learned features are more robust. Additionally, ReLU is used like an activation function non-saturating to address the vanishing gradient problem and improve the convergence rate. Overlapping subsampling and local response normalization are applied to improve generalization by reducing overfitting. Further enhancements include using larger filters (5×5 and 11×11) in earlier layers to improve upon previous network performance. AlexNet has played a pivotal role in the evolution of recent Convolutional Neural Network (CNN) architectures and has ushered in a groundbreaking era of research into CNN-based applications

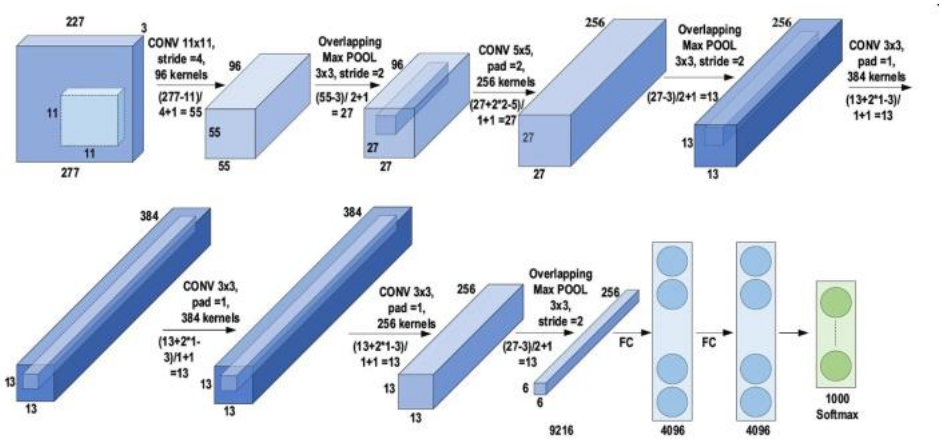

Figure 2.9. Alexnet architecture

2.7.2. ZefNet

Before 2013, the CNN learning method relied primarily on trial and error, lacking a precise understanding of its specific purpose after improvement. This limitation hindered the deep CNN's performance in interpreting complex images. In 2013, Zeiler and Fergus (Zeiler & Fergus, 2014) addressed this issue by introducing the multilayer deconvolutional neural network (DeconvNet). This approach eventually came to be recognized as ZefNet, which was created to visually represent the network quantitatively. The aim of visualizing the network activity was to monitor the performance of CNN by comprehending neuron activation. DenconvNet functions like a forward-pass CNN by reversing the sequence of operation between convolutional and pooling layers. This type of reverse mapping sends the output from the convolutional layer backwards, producing visually perceivable image shapes that correspondingly provide insight into the internal feature representation neural interpretation learned at every layer. Throughout the training stage, ZefNet focused on monitoring the learning process and identifying any issues with the model. The experimental application of DeconvNet to AlexNet revealed that only specific neurons were active in the first two layers of the network while others were inactive. It further showed that the extracted features from the second layer comprised aliasing objects, prompting Fergus and Zeiler to modify the topology of CNN and optimize parameters. They adjusted stride and filter sizes, leading to improved performance. This rearrangement highlighted how feature visualization could identify design weaknesses and guide parameter adjustments (see **Figure 2.10**).

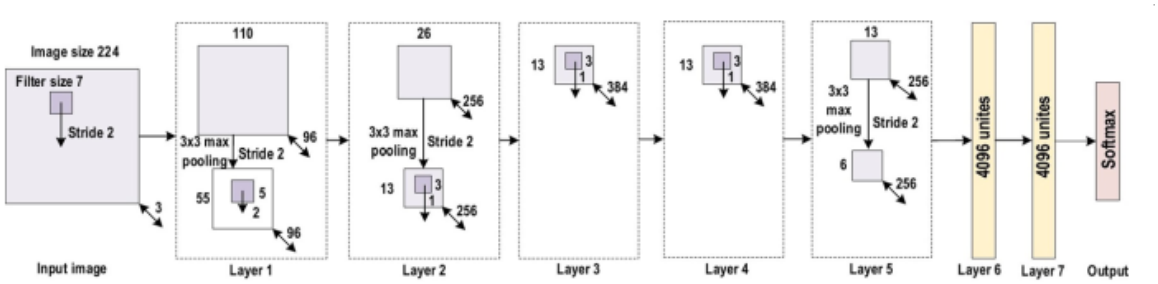

Figure 2.10. ZefNet architecture

2.7.3. VGG (Visual Geometry Group)

After CNN's effectiveness in image recognition was established, Simonyan and Zisserman proposed a straightforward and efficient design principle for CNN known as the Visual Geometry Group. A multi-layered (Simonyan & Zisserman, 2014) structure, it included nineteen additional layers compared to ZefNet (Zeiler & Fergus, 2014) and AlexNet (Krizhevsky et al., 2012) in order to mimic the structural capacity of the network at varying depths. On the contrary, during the competition of 2013- ILSVRC, ZefNet was considered an innovative network which suggested smaller filters could improve CNN performance. VGG introduced a new approach by incorporating 3×3 filters instead of the larger 5×5 and 11 × 11 filters used in ZefNet. This experimental demonstration revealed that utilizing these smaller-size filters could yield similar results to using larger-size filters, making the receptive field equally effective. Furthermore, employing small-size filters led to a reduction in computational complexity by decreasing the number of parameters. These findings have sparked a novel research direction for integrating diminutive filter structures into CNNs. Additionally, VGG managed network complexity by introducing 1×1 convolutions within the convolutional layers, enabling it to learn linear combinations of subsequent feature maps. In terms of network optimization, the layer of max pooling is added after a convolutional layer, and padding is used for preserve spatial resolution. The VGG has demonstrated notable performance in addressing localization issues and classifying of images. Although this architecture did not secure the first position in the competition of 2014- ILSVRC, its increased depth, uniform structure, and simplicity earned it recognition. However, VGG's computational overhead was high because it relied on approximately 140 million parameters, which was considered its primary drawback. Refer to **Figure 2.11** for an illustration of the network architecture.

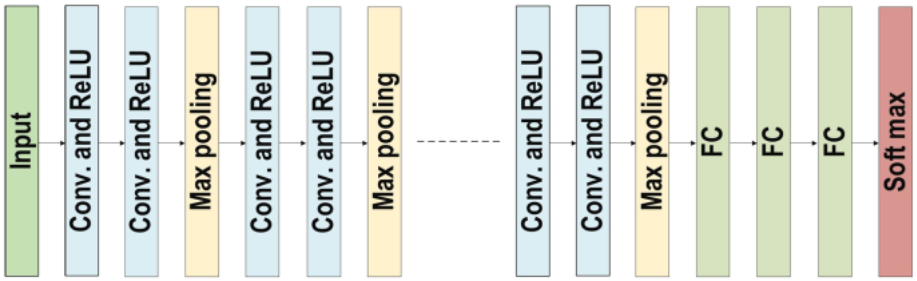

Figure 2.11. VGG architecture

2.7.4. GoogleNet

GoogleNet, also known as Inception-V1, emerged victorious (Szegedy et al., 2015) in the 2014-ILSVRC competition by focusing on achieving high accuracy while reducing computational costs. The GoogleNet architecture introduced a new concept of an inception block (module) within the context of CNNs, which integrates multiple-scale convolutional transformations using merge, transform, and split functions for feature extraction. **Figure 2.12** depicts the structure of the inception block. This design incorporates filters of varying sizes (5×5, 3×3, and 1×1) to capture channel information along with spatial information at different ranges of spatial resolution.

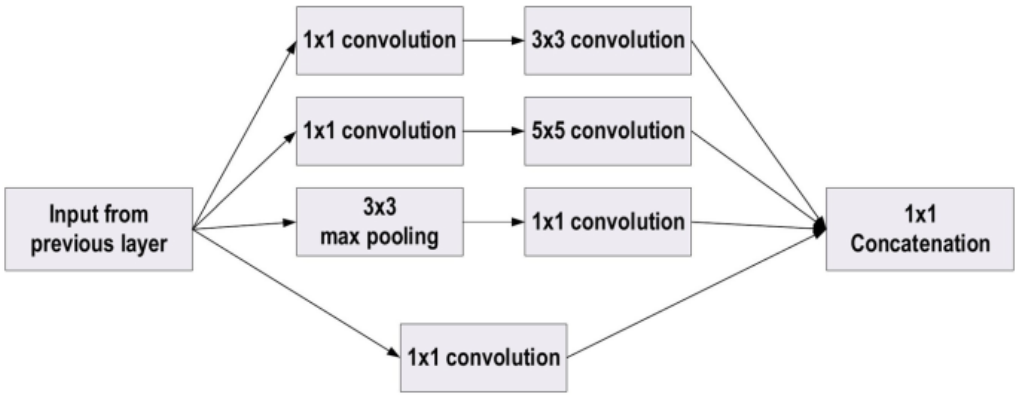

Figure 2.12. Google Block's basic architecture

The Google Net utilized concepts of transform, split and merge to address issues related to different existing variants in similar category of images. It aimed to improve the CNN parameters effectiveness and enhance learning capacity by using 1×1 convolutional filters as bottleneck layers before employing large- size kernels. The sparse connections were employed to address the challenges posed by redundant information problems and decrease

costs by neglecting irrelevant channels. The GAP layer was used in the last like the end layer instead of the Fully Connected layer, reducing the connections density and significantly decreasing the parameters number from 40 millions to 5 millions due to the tunings parameter. In addition, the factors of regularity included employing Rms Prop as a batch normalization and an optimizer. Google Net proposed the auxiliary learners use to accelerate convergence. However, its heterogeneous topology presents a challenge in adapting from one module to another. Moreover, Google Net suffers from representation jam, that results in feature space decreased and occasional loss of valuable information in subsequent layers.

2.7.5. ResNet

He et al. (He, Zhang, Ren, & Sun, 2016) developed ResNet, the winner of ILSVRC 2015, to address the vanishing gradient issue in deep networks. ResNet different versions were created based on layers number, with ResNet50 being the most common type consisting of 49 convolutional layers and a single FC layer. The network had 25.5 M weights and required 3.9 M MACs for computation. A ResNet key feature is its bypass pathway concept, as illustrated in **Figure 2.13.** This concept is depicted in **Figure 2.13**, featuring the core structure of the ResNet block diagram. It consists of a traditional feedforward network with an additional residual connection. The output of the residual layer corresponds to the (l-1)th outputs from the preceding layer (x_l-1). Opting for various operations such as convolution utilizing variable-size filters or batch normalization before applying an activation function like ReLU on (x_l-1), produces the output F(x_l-1). This leads to x_l as the final residual output, which can be expressed mathematically using Equation (2.9).

$$x_l = F(x_l - 1) + x_l - 1 \qquad (2.9)$$

Numerous fundamental residual blocks are present in the structure of residual network. Depending on specific architecture of residual network, modifications to operations within the residual block are occur.

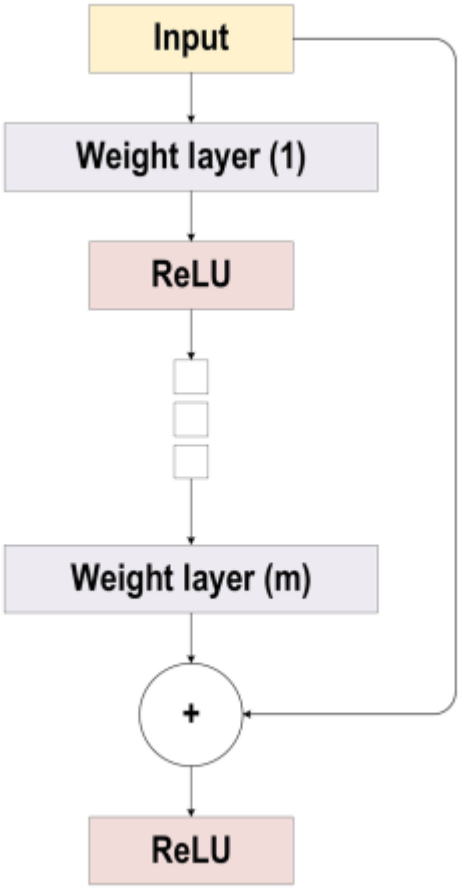

Figure 2.13. ResNet block diagram

2.8. Conclusion

We have presented, throughout this chapter, definitions and basic concepts of artificial intelligence, machine learning and deep learning. We explained the relation between machine learning and deep learning. Then, we review some types of deep learning networks, the CNN structure, and the most frequent methods for object classification using deep learning. In addition, we cited common CNN model architectures.

In the next chapter, we will review the existing works in the literature in the field of location recognition and image geo-localization by citing different existing techniques and methods. Related work can be split into two categories; image retrieval-based methods and those classification-based. However, there are a set of techniques called Structure-from-Motion (SfM). In addition, approaches based on data from multiple modalities and deep learning are also mentioned.

CHAPTER 3

LITERATURE REVIEW

CHAPTER 3: LITERATURE REVIEW

3.1. Introduction

In this chapter, we present the basic concepts and review the prominent approaches in the field of image geo-localization, also cite different existing techniques and methods introduced in the literature. Generally, related work can be approximately splited into two categories; image retrieval-based and classification-based techniques. However, researchers developed approaches for localization and camera orientation (camera pose estimation) from only one image and camera pose tracking from a continuous sequence of images. This set of techniques is called Structure-from-Motion (SfM) where many works have been provided. For this reason, we mentioned them under the title of approaches based on structure from motion. Due to methods that use other input data like ortho photomaps combined with attribute maps, bird's eye or satellite weather imagery to find camera location for a query image and tremendous spread of deep learning techniques; approaches based on data of multiple modalities and deep learning are cited. Moreover, we review the most used dataset in this field for applications and measures.

3.2. Image based-localization

Geo-localization is the general term used to describe the process of identifying the geographical location of an entity, usually expressed as GPS coordinates. This entity could be various things such as an image, a series of images, a video, a satellite image, or image objects.

Image geo-localization, also known as image-based localization or visual place recognition, presents a significant challenge in the fields of learning and computer vision, as it seeks to identify or estimate and predict the geographic location depicted in a picture based solely on its visual information. It can be also considered as the task of determining the Global Positioning System (GPS) coordinates of where a photo was taken as precisely as possible. Despite substantial progress in computer vision, this remains a complex issue that has been under intense research scrutiny over recent years. The surge in publicly available images and datasets on the Internet has further fueled investigations into predicting the geographic locations of images.

Images are closely connected to their locations (Amir R. Zamir, Hakeem, Van Gool, Shah, & Szeliski, 2016). Hence, numerous such applications necessitate the establishment

of a two-way connection between the visual content and the geographical location. This problem requires careful analysis of the content. Therefore, having knowledge about the specific location where an image was captured is a crucial attribute of that image. This creates a high demand for thorough content analysis in such cases. Zamir and Shah (Amir Roshan Zamir, 2014) describe the geo- localization as "…estimating the geo-location of a query image by finding its matching reference images,…". The authors of this paper (Mayank Bansal, Sawhney, Cheng, & Daniilidis, 2011) said "Given a ground-level street view (SV) image in an urban area, we want to determine the geo-location of the camera in the absence of any metadata (GPS or camera parameters)". Hays et al. in his work (Hays & Efros, 2008) describes the geo-localization issue as ". . . estimating a distribution over geographic locations from a single image . . . ,". Additionally, Seo et al. (Seo, Weyand, Sim, & Han, 2018) said "Image geolocalization is the task of predicting the geographic location of an image based only on its pixels without any meta-information". Glistrup et al. (Glistrup, Rudinac, & Jónsson, 2022) define image geo-localization as "… geo-tagging, geolocating, geospatial localization, location inference, or location estimation, is the task of estimating or inferring the real-world location in which an image was taken…". Masone et al. (Masone & Caputo, 2021) consider the visual place recognition as "… the task of recognizing the place depicted in an image (or a sequence of images)… ", where the definition of place can vary depending on the task, for example it could be identified by a landmark's name, GPS coordinates, or a 6 DoF pose with respect to a frame of reference.

In the literature, two main approaches have been proposed to handle the problem of image geo-localization: retrieval-based and classification-based methods. However, researchers developed other techniques to tackle this issue, like structure from motion (SfM)-based, data of multiple modalities-based and deep learning-based approaches. All of these methods are mentioned below.

3.2.1. Approaches based on image retrieval

The most commonly used approach for image geo- location relies on image retrieve techniques. Hays et al. (Hays & Efros, 2008, 2015) explore the task of global image geo-localization to estimate where an image was captured on the Earth. They examine kinss of the "im2gps" algorithm utilizing millions of "geo-tagged" Internet images as training dataset. Initially, the authors discussed the nearest-neighbor baseline. Next, they present a lazy- learning method with more advanced features that enhance the performance of the original "im2gps" approach. Beyond quantifying the accuracy of geo-localization. In

addition, they resolve (1) how the non-uniform training data distribution impacts the method (2) how performance compares to baselines like land-cover recognition and random guessing, and (3) whether geo-localization is simply landmark or “instance level” recognition at a large scale. Furthermore, they prove that geo-location estimation can provide the basis for image recognition tasks such as land cover or population density estimation.

Vo et al. (Vo, Jacobs, & Hays, 2017) suggest merging this method with the initial Im2GPS approach, where a query image is compared to a geotagged image database and its location is deduced from the matched set. Their estimation of the geographical position of a query image involves using kernel density estimation on the locations of its closest neighbors in the reference database. Interestingly, they discover that the most effective attributes for their retrieval task are obtained from networks trained with classification loss, even though they do not employ a classification approach during testing. Their straightforward method achieves top-tier accuracy in geolocation while also necessitating substantially less training data

In (Arandjelovi et al., 2016), Arandjelovic et al. make three main contributions. Firstly, they create a Convolutional Neural Network architecture designed for direct end-to-end training specifically for the place recognition task. Central to this architecture is NetVLAD, a novel generalized VLAD layer inspired by the widely used "Vector of Locally Aggregated Descriptors" image representation in image retrieval. This layer can be easily integrated into any CNN architecture and trained using back propagation. Secondly, they introduce a new weakly supervised ranking loss-based training procedure to train the architecture's parameters directly from images showing the same places over time obtained from Google Street View Time Machine data. Lastly, their study demonstrates that the proposed architecture outperforms non-learned image representations as well as off-the-shelf CNN descriptors on two challenging place recognition benchmarks while also improving upon current state-of-the-art compact image representations on standard image retrieval benchmarks.

Global characteristics have the benefit of retrieving a wide range of natural scene images by utilizing surrounding details, while local image features offer higher precision in identifying structured objects such as buildings and are consequently more commonly utilized (Cao & Snavely, 2015; D. M. Chen, Baatz, & Tsai, 2011; Kim, Dunn, & Frahm, 2015; Knopp, Sivic, & Pajdla, 2010; Schindler, Brown, & Szeliski, 2007; Amir Roshan

Zamir & Shah, 2014). Noh et al. (Noh, Sim, & Weyand, 2018) present a novel attentive local feature descriptor named DELF for large-scale image retrieval. Based on convolutional neural networks, it is trained using only image-level annotations from a landmark image dataset. Introducing an attention mechanism for keypoint selection, the descriptor shares most network layers with the selective process, thereby identifying semantically useful local features. This framework can replace other keypoint detectors and descriptors to enable more precise feature matching and geometric verification in image retrieval tasks. To assess the proposed descriptor's performance, they introduce the Google-Landmarks dataset —a new large-scale dataset that poses challenges such as background clutter, partial occlusion, multiple landmarks, and objects in various scales among others both in database and query scenarios They demonstrate that DELF achieves significantly superior results compared to state-of-the-art global and local descriptors in this extensive setting.

Another correlated field of study involves landmark recognition (Lin, Belongie, & Hays, 2013; Workman, Souvenir, & Jacobs, 2015; Amir Roshan Zamir & Shah, 2014). The images are categorized by their visual resemblance and geo-locations to create widespread landmarks database. Zamir et al. (Amir Roshan Zamir & Shah, 2014) propose a novel approach for geolocating an image by introducing a new method for matching multiple nearest neighbors using Generalized Minimum Clique Graphs (GMCG). Initially, they derive local characteristics such as Scale-Invariant Feature Transform (SIFT) from the query photo and retrieve several closest neighbors for each query characteristic from the dataset of reference. Subsequently, they utilize their characteristic matching based on GMCP to choose a sole closest neighbor for each query feature to ensure global consistency across all matches. Their method challenges the notion that first nearest neighbors may not always be optimal choices in image correspondence. Instead, it takes into account several nearby reference points as possible matches and determines the appropriate ones by ensuring uniformity among their overall characteristics such as GIST using GMCP. The authors argue that employing a robust distance function is crucial when dealing with scenarios where query matches several dissimilar reference images based on global characteristics; thus, they suggest a reliable Gaussian Radial Basis Function-based distance function. They assessed this structure on a novel street view dataset that comprised 102 000 photos and demonstrated its performance surpasses existing methods by 10 percent in experimental evaluations.

I image retrieval system, the dataset is worked as an index (Avrithis, Kalantidis, Tolias, & Spyrou, 2010; Weyand & Leibe, 2015) or for training the landmark classifier (Bergamo & Torresani, 2013; Gron, Obozinski, & Sivic, 2013; Li, Crandall, & Huttenlocher, 2009).

3.2.2. Approaches based on classification

In the field of computer vision, the issue of image geo-localization is commonly addressed through image retrieval techniques. However, Weyand et al. (Weyand, Kostrikov, & Philbin, 2016) take a different approach by framing the issue as a task of classification where they divide the surface of earth into numerous geographic cells at multi-scale and then train the deep network utilizing millions of geo-tagged photos. Unlike prior methods that only recognize landmarks or utilize global image descriptors for approximate matching, their model can leverage and combine multiple visual cues. Their study demonstrates that their resulting model, known as PlaNet, surpasses prior approaches and in some instances achieves accuracy levels exceeding human capabilities. Furthermore, they expand their algorithm to encompass image albums by integrating it with a long short- term memory architecture. Through learning how to exploit temporal coherence in order to geolocate ambiguous photos, the authors illustrate that this combined model delivers a 50% improvement in performance compared to the single-image model.

Subsequent studies have further enhanced classification-based geo-localization through the use of advanced models, refinement of cell partitioning methods, and utilization of expanded datasets. Seo et al. (Seo et al., 2018) present a new model to partition cells for image geo-localization using a classification-based approach and combinatorial partitioning to define detailed class configuration by combining diverse geographical sets at broader levels. They propose an efficient algorithm, combinatorial partitioning, which creates numerous fine-grained output classes through intersecting multiple coarse-grained partitions of the earth. Each classifier supports the fine-grained classes that intersect with their respective coarse-grained ones, enabling accurate location prediction while ensuring adequate training examples for each class.

3.2.3. Approaches based on structure from motion (SfM)

Structure-from-Motion encompasses a collection of methods for generating a 3D representation of a scene based on multiple 2D images capturing the same scene from different angles. By establishing the connection between 2D pictures and the resulting 3D model, researchers have been able to devise procedures for determining location and camera

alignment (camera pose estimation) from an individual image as well as tracking camera poses across a continuous sequence of images. The resultant 3D model comprises positions in space and orientations of both the cameras used to capture the original imagery and points in three-dimensional space. One unique benefit is that these datasets can yield highly accurate reconstructions of camera poses within queries, with minimal margin for error, down to units measured in meters.

In (Cao & Snavely, 2015; Irschara, Zach, Frahm, & Bischof, 2009; Li, Snavely, Huttenlocher, & Fua, 2016; Sattler et al., 2012; Sattler, Leibe, & Kobbelt, 2011), approaches based on pose estimation match query images against 3D models of a region, and use 2D-3D feature correspondences to determine 6-DOF query poses. Sattler et al. (Sattler et al., 2012) examine the algorithmic elements contributing to performance differences, identifying false positive votes as the primary factor. Through a thorough experimental assessment, they demonstrate that retrieval techniques utilizing selective voting surpass current direct matching methods. The authors also investigate acceleration possibilities for both correspondence computation and selective voting by employing feature descriptors Hamming embedding. Additionally, they present a novel challenging query images dataset to assess image-based localization.

Based on a convolutional neural network, PoseNet (Kendall & College, 2016) is deep convolutional neural network that is designed to predict the 6-degree-of-freedom (6-DOF) camera pose from a single image. It achieves localization using high-level features and can handle challenging lighting conditions, motion blur, and unknown camera properties where traditional point-based SIFT registration methods struggle. However, its initial training utilized a simplistic loss function with hyperparameters that necessitate expensive fine-tuning. In their work, Kendall et al. (Kendall & Cipolla, 2017) approach this issue with a more comprehensive theoretical framework. They introduce various innovative loss functions for learning camera poses based on geometry and scene reprojection error. Additionally, they demonstrate an automated method for determining optimal weights to simultaneously estimate position and orientation. By leveraging geometric principles, the authors illustrate significant performance improvements of their technique compared to PoseNet across diverse datasets including indoor environments and urban scenes.

Walch et al. (Walch, Leal-taix, Sattler, & Cremers, 2017) introduce a novel CNN+LSTM structure aimed at improving camera pose regression for various indoor and outdoor scenes by employing an intermediate LSTM layer in the network to reduce dimensionality, resulting in substantial enhancements in localization performance. The authors conduct a comprehensive quantitative comparison of CNN-based and SIFT-based localization techniques, highlighting the strengths and weaknesses of each approach. Additionally, they release a large-scale indoor dataset with precise ground truth obtained from laser scanning. Through experimentation on public datasets encompassing both indoor and outdoor settings, their method demonstrates superior performance compared to existing deep architectures, effectively localizing images even under challenging conditions such as textureless surfaces where traditional SIFT-based methods falter.

3.2.4. Approaches based on multiple modalities data

Unlike the approaches mentioned above, methods that leverage data from various modalities use additional input data to find camera location for a query image. Methods using other input data like ortho photomaps combined with attribute maps, and bird's eye or satellite weather imagery were exploited. In this domain, mainly methods for outdoor, non-urban environments are considered. This is due to the nature of areas with lower density of population – for image-based methods there is not so much ground-level imagery, so other sources are used.

Cross-view geo-location utilizes satellite and aerial images to identify query locations. A popular choice is the cross-domain matching of a query image and a terrain model, with the utilization of features like horizon lines, ridges, and edge maps. Baatz et al. (Baatz, Saurer, Köser, & Pollefeys, 2012) present an automated method for extremely large-scale visual localization that can effectively utilize both visual information (contours) and geometric constraints (consistent orientation) simultaneously. They validate the system on the scale of a whole country (Switzerland, 40000 km^2) using a new dataset of more than 200 landscape query pictures with ground truth.

Lin et al. (Lin et al., 2013) have proposed a comprehensive cross-domain matching framework to significantly expand the reach of image geo-location. The majority of the Earth's surface lacks publicly available ground-level geo-tagged photos, rendering traditional image-to-image matching methods ineffective for queries in such areas. Through their new dataset consisting of ground level, aerial, and attribute images, they assessed the

performance of various baseline and innovative approaches for "cross-view" geo-location. Their "discriminative translation" approach involves training an aerial image classifier based on ground-level scene matches and has demonstrated the ability to roughly geo-locate 17% of isolated query images. While their experiments were conducted on a modest scale (1600 km^2 region), their approach is easily scalable in terms of data availability and computational complexity. This method marks the first time overhead imagery and land cover survey data are utilized for photograph geolocation purposes, complementing the mountain-based geo-location method introduced by (Baatz et al., 2012), which utilizes widely available geographic features like digital elevation maps.

In (Workman et al., 2015), Workman et al. suggest utilizing deep convolutional neural networks to tackle the challenge of cross-view image geo-localization, where a ground-level query image's geographic location is estimated by matching it with geo-referenced aerial images. They leverage advanced feature representations for ground-level images and propose a cross-view training method to learn a unified semantic feature representation for aerial images. Additionally, they introduce a network architecture that integrates features extracted from aerial images at various spatial scales. To facilitate the training of these networks, the authors create an extensive database containing pairs of aerial and ground-level images from diverse locations across the United States. Their approaches demonstrate superior performance compared to existing methods on two standard datasets. Furthermore, they illustrate qualitatively that the proposed feature representations exhibit discriminative qualities at both localized and continental spatial levels.

By using CNN for the cross-view matching, Lin et al. (Tsung-Yi Lin, Yin Cui, Serge Belongie, 2015) use Google Street View and aerial "bird's eye" photos, which are taken at an angle compared to traditional aerial orthophotos captured perpendicular to the terrain, researchers identify the query image of ground-level by matching it to aerial images reference database. They utilize an available images to construct a dataset comprising 78000 aligned pairs of cross-view image. The main challenge lies in the fact that conventional computer vision methods struggle with the appearance variations and wide baseline present in these pairs of cross-view. Leveraging their dataset, they develop representation of feature that brings matching views close together while separating mismatched ones. Their proposed Where-CNN approach is influenced by the success of deep learning in verification of faces and delivers remarkable advancements over both handcrafted features and deep features

obtained from different large-scale databases. The authors demonstrate the Where-CNN efficiency in identifying matches between aerial imagery and street view while showcasing its capability to generalize features learned across new locations.

Tian et al. (Y. Tian, Chen, & Shah, 2017) conducted research similar to that of Lin et al. (Tsung-Yi Lin, Yin Cui, Serge Belongie, 2015), where they utilized Google Street View query images and aerial "bird's eye" reference imagery for estimating the GPS location of the query image by identifying matching images in the geo-tagged reference dataset "bird's eye" view images, and vice versa. In this study, a new framework for cross-view imagery geo- localization is introduced, leveraging deep convolutional neural networks known for their success in classifying of images and detecting of objects. Initially, the Faster R-CNN is employed to detect buildings in both query and reference images. Subsequently, utilizing a Siamese network trained on positive matching image pairs as well as negative pairs, k nearest neighbors are retrieved from buildings in reference for each query building in the image. To determine the correct nearest neighbor for each query building, an efficient multiple nearest neighbors matching method based on dominant sets is devised by them. The effectiveness of the proposed framework is evaluated using a novel dataset comprising street view and bird's eye view image pairs. Experimental findings demonstrate that their method outperforms other approaches regarding geo-localization accuracy while also demonstrating generalizability to unseen locations.

3.2.5. Approaches based on deep learning

Deep learning is a recently developed method designed to produce impressive outcomes and illustrate its ability to generate strong features in various computer vision tasks. It captures advanced features derived from entire image components where detailed characteristics using deep learning, such as Convolutional Neural Networks, can be directly acquired from images due to their greater effectiveness compared to traditional handcrafted features.

Convolutional Neural Networks are prime illustrations of how neural networks can acquire the ability to recognize features and patterns from images, enabling them to undertake complex tasks such as image classification and object recognition.

In recent times, deep learning techniques have led to improved accuracy in location recognition, paving the way for new possibilities in computer vision applications that rely on detailed image representations. Consequently, researchers have effectively utilized

various CNN architectures to predict the locations depicted in images (Johns, Rounds, & Henry, 2017; Muller-Budack, Pustu-Iren, & Ewerth, 2018; Peddada & Hong, 2015; Tsung-Yi Lin, Yin Cui, Serge Belongie, 2015; Workman et al., 2015) and recognize places and locations (Z. Chen et al., 2017; Crandall, Li, Lee, & Huttenlocher, 2016; Suresh, Chodosh, & Abello, 2018; Y. Tian et al., 2017; Q. Wang, Zhou, & Xu, 2017; Weyand et al., 2016).

Müller-Budack et al. (Muller-Budack et al., 2018) tackle the issue of estimating geo-location with multi- partitioning method that integrates hierarchical knowledge across various spatial resolutions. Additionally, they extract and gather data on diverse environmental types such as urban settings, indoor and natural. The authors approach image geo-location as a classification challenge by segmenting the earth into geographical cells containing an equal images number, similar to the methodology employed in PlaNet's work (Weyand et al., 2016).

Johns et al. (Johns et al., 2017) demonstrate the Delaunay triangles efficacity as a type of mesh for geo-localization in scenarios with low volume relatively. Their research contributes to estimating image geo-locations by introducing a new global meshing approach, outlining various training techniques to overcome limitations in data volume for model training, and showing how incorporating additional information can enhance the effectiveness of geo-location inference model.

In (Workman et al., 2015), the authors employ advanced convolutional neural networks to tackle the challenge of cross-view image geo-localization, involving the matching of ground-level queries with aerial imagery for localization. Furthermore, they introduce a network structure that integrates extracted characteristics from the aerial images across varying spatial scales.

The authors (Tsung-Yi Lin, Yin Cui, Serge Belongie, 2015) propose a method called Where-CNN, drawing inspiration from the effectiveness of deep learning in verification of faces, to enhance the ground-level query images geo-location by comparing them with a aerial images reference database. This approach presents the geo-localization of images as an identity verification task. In the context of image geo-location, methods of image retrieval are frequently employed. However, the researchers in (Weyand et al., 2016) approach the issue as a classification task by partitioning the planet into numerous multi-scale geographic cells. They then use millions of geo-tagged images to train a deep network. The model,

known as PlaNet, demonstrated superior performance compared to previous approaches and achieved exceptionally high levels of accuracy in certain instances.

In (Y. Tian et al., 2017), Tian et al. address the cross-view image geo-localization challenge, aiming to predict the GPS location of a queried street view image by identifying its corresponding images in geo-tagged bird's eye view images database, and vice versa. The authors propose an innovative framework for cross-view image geo-localization that leverages the success of deep convolutional neural networks in tasks such as image classification and object detection. Initially, they employed Faster R-CNN to identify buildings in both query and reference images. Subsequently, they utilize a Siamese network trained on matching pairs of positive and negative images to retrieve the k nearest neighbors from the reference buildings for each building present in the query image. Lastly, employing dominant sets theory, they devise an efficient method for multiple nearest neighbor matching to accurately determine suitable matches for each building within the query set.

Wang et al. (Q. Wang et al., 2017) applied deep learning techniques to categorize images of Street View by conducting a study to determine the most suitable convolutional neural network model for this task. They gathered their own dataset and tailored the structure and training methods of the AlexNet convolutional neural network model to fit the characteristics of the dataset. The experiments involved using max-pooling as a sampling model, setting 128 samples per iteration, and randomly cropping different parts of the same picture for input into the model, resulting in a high accuracy rate. The findings demonstrated that increasing the number of samples per iteration significantly improved the training effectiveness of their proposed model.

In (Suresh et al., 2018), Suresh et al. examine the challenge of identifying the geographical location of an image, treating it as a classification problem to determine the most likely state among 50 in the USA. They introduce and share the 50States10K dataset, comprising 0.5 million Google Street View images from across the country. Utilizing a deep neural network rooted in ResNet architecture, they train the model and outline four distinct methods for integrating low-level cardinality information. As a result, their model demonstrates test dataset accuracy that is significantly greater than chance by a factor of 20.

Chen et al. (Z. Chen et al., 2017) conducted extensive training of two CNN architectures on a large scale for the specific task of place recognition. They utilized a multi-scale feature encoding approach to generate features that are invariant to changes in conditions and viewpoints. To facilitate this training, they curated a substantial Specific Places Dataset containing hundreds of examples showing variations in place appearances at thousands of different locations. This new dataset allowed them to establish a training framework that treats place recognition as a classification problem. The researchers thoroughly assessed their trained networks using various challenging benchmark datasets for place recognition and demonstrated an average 10% improvement in performance compared to other algorithms for place recognition and pre-trained CNNs. By analyzing the network responses and comparing them with those from pre-trained networks, they provided valuable insights into what is learned by the network during training for place recognition, highlighting its implications for future research in this field.

The significant expansion of social media platforms in recent years has resulted in vast repositories of online images, presenting new difficulties in efficiently organizing them. A particularly instinctive method for browsing and searching images is by their geographical location, but most online images lack GPS metadata linked to them. Crandall et al. (Crandall et al., 2016) addressed the challenge of identifying well-known landmarks in large collections of diverse consumer images. They approached this by creating a classification task with around 2 million images and 500 categories, which were automatically formed from geo-tagged photos on Flickr. The authors developed models for these landmarks using multiclass support vector machines and utilized interest point descriptors as features. Furthermore, they integrated non-visual metadata from photo-sharing sites, demonstrating that textual tags and temporal constraints significantly improved the accuracy of landmark recognition. Additionally, they explored Convolutional Neural Networks within deep learning techniques and found them to outperform traditional recognition methods considerably, surpassing human observers in certain scenarios.

3.3. City-scale image geo-localization

Image geo-localization is often simplified by focusing on city-scale recognition of landmarks and cities. Schindler et al. (Schindler et al., 2007) explore location recognition in a large image dataset utilizing a vocabulary tree, aiming to determine a query image location within a substantial dataset containing street-side images of the city. They investigate the

challenges posed by traditional invariant feature matching methods as the size of the database increases. Meanwhile, Zheng et al. (Zheng et al., 2010) aim to model and recognize landmarks at a global scale, leveraging multimedia data available on the web, Internet image search engines, and advancements in object recognition and clustering techniques. In their work (Arandjelovic, Gronat, Torii, Pajdla, & Sivic, 2018), the authors address large-scale visual place recognition, with an emphasis on quickly and accurately identifying the location of given query photographs.

Studies have been conducted on natural landscapes such as mountains and deserts. Baatz et al. (Baatz et al., 2012) focused on mountainous terrain, utilizing digital elevation models to efficiently extract representations for quick visual database lookup. They proposed an automated method for large-scale visual localization that simultaneously leverages visual information (contours) and geometric constraints (consistent orientation). Similarly, Saurer et al. (Saurer, Baatz, Köser, Ladický, & Pollefeys, 2016) concentrated on camera geo-localization in natural settings, particularly emphasizing skyline recognition in a query image with the use of a digital elevation model. The last two papers discussed mountain recognition while desert imagery was the focus of this work (Tzeng, Zhai, Clements, Townshend, & Zakhor, 2013). The authors presented a system for user-aided visual localization of desert imagery without relying on metadata such as GPS readings or camera specifications; only publicly available digital elevation models were used to accurately locate photographs in non-urban environments like deserts.

3.4. Large-scale image geo-localization

Only a few methods address the issue at global-scale. Hays et al. (Hays & Efros, 2008)(Hays & Efros, 2015) explore the task of global image geo-localization by estimating where on the Earth a photograph was captured. They attempt to resolve the global geo-localization problem by using Internet-derived training data.

Vo et al. (Vo et al., 2017) try to combine their approach based on the IM2GPS approach where they estimate the geographic location of a query image by applying kernel density estimation to the locations of its nearest neighbors in the reference database.

In (Weyand et al., 2016), Weyand et al. pose the problem as one of the classification tasks by subdividing the surface of the earth into thousands of multi- scale geographic cells, and train a deep network using mil- lions of geo-tagged images. Previous techniques only identified landmarks or conducted rough matching with global image descriptors, but their

model can incorporate multiple visible cues. It demonstrates that the resulting PlaNet model surpasses previous methods and can even achieve accuracy levels beyond human capability in certain scenarios.

3.5. Data sets

Research and investigations into place recognition and image geo-location issues rely on a variety of datasets, typically utilized to assess and contrast current and updated methodologies. Numerous datasets can be accessed from online sources like Google Street View, Panoramio, or Flickr for urban areas. Conversely, the availability of datasets for geo-localization in natural settings such as mountains is limited.

In this part, we present the commonly utilized datasets in global, urban, and natural settings. These datasets are generally accessible for free download. Table 3.1 provides an overview of the mentioned datasets categorized based on their respective environments.

3.5.1. Google Street View Dataset

Zamir and Shah's dataset (Amir Roshan Zamir & Shah, 2014) comprises 62,058 high-resolution Google Street View photos captured in Pittsburgh, Orlando, and partly Manhattan. Each street segment features complete 360° panoramic images divided into four side views and one top view image. This collection is beneficial for precise position determination and camera orientation assessment. Figure 3.1 illustrates certain samples from both the reference dataset and test set

3.5.2. WorldCities dataset

This publicly accessible dataset was gathered and presented as a new collection of data in the research by Zemene *et al.* (Zemene et al., 2019). The WorldCities dataset comprises 300,000 high-resolution Google street view images encompassing 14 diverse cities across the globe, including Amsterdam, Frankfurt, Paris Rome, Milan, Melbourne and Sydney, Los Angeles, Chicago, Las Vegas, Phoenix, San Diego, Houston Dallas. It presents more complex challenges compared to other datasets due to the diversity in building similarity worldwide encompassing aspects such as color variations, edges shapes and wall designs. This is depicted in Figure 3.2.

Figure 3.1. Left: Examples from street view images from the reference set. Right: Examples from user-uploaded images from the test set.

Figure 3.2. The top four lines are examples of street view images from eight different places. The bottom two lines are examples of user-uploaded images from the test set.

3.5.3. The New Data in Multimedia Research

It called also the Yahoo image dataset and its abbreviation is YFCC100M. This dataset was published by Thomee *et al*. (Thomee, Shamma, Friedland, Poland, & Borth, n.d.) which contains hundred million datasets from Yahoo Flickr images and videos see Figure 3.3. It is under Creative Commons licenses, that make the data easily usable for anyone. The metadata provided are tags, times pan, GPS locations and camera information.

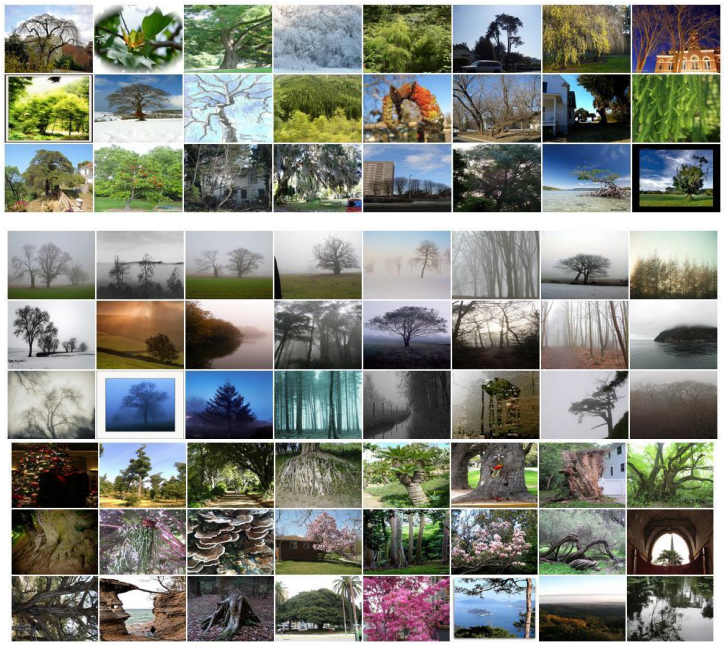

Figure 3.3. Some examples images from YFCC100m dataset

Figure 3.4. Examples from the new benchmark Alps100K.

3.5.4. Alps100K dataset

This dataset is presented as a new dataset of about 100 000 images from mountain environments which cover a vast geographic area of the Alps, see Figure 3.4. It was collected by Čadík *et al.* (Cadík & Vaší˘, 2015), also it has information from Flickr which contains GPS coordinates which make it able to be used for geo-localization challenges.

3.5.5. San Francisco Landmark Dataset

The San Francisco dataset includes 1.06 million database images obtained from approximately 150,000 panoramic images taken by surveying vehicles on the streets of San Francisco. The 803 query images were captured using various cell phones. Each database

image is labeled with a "carto id" representing the visible building in the image, and a list of relevant "carto ids" is supplied for each query image. This dataset was presented by Chen *et al.* (D. M. Chen et al., 2011). Sample visuals from the dataset are illustrated in Figure 3.5.

Figure 3.5. Samples images of San Francisco dataset.

3.5.6. IM2GPS dataset

Introduced by Hays and Efros (Hays & Efros, 2008). This dataset contains GPS coordinates for each image. The photos in this data collection are predominantly taken in urban areas, with a notable preference for capturing famous landmarks such as the Eiffel Tower.

3.5.7. VPRiCE dataset

The dataset originates from the Visual Place Recognition in Changing Environments 2015 challenge (VPRiCE) ("The vprice challenge 2015 visual place recognition in changing environments," n.d.) and is used for location retrieval. It comprises *memory* and *live* images of outdoor environments captured under different viewing conditions. The *memory* dataset contains images observed during the initial visit, serving as a reference against which images from the *live* dataset are matched. There may be significant changes in environmental conditions between the two traverses. Overall, the dataset includes 7778 images at a resolution of 640x480, recorded from trains, cars, buses, bikes, or pedestrians as depicted in Figure 3.6.

Figure 3.6. Examples of images from VPRiCE dataset.

3.5.8. Pittsburgh dataset

The images in the Pittsburgh dataset (Torii & Pajdla, 2013) were acquired by selecting 254,000 perspective images from approximately 10,600 panoramas sourced from Google Street View. Additionally, around 24,000 query images were obtained from a separate set of Google Street View panoramas within Google's Pittsburgh Research Dataset. Both sets of panorama images possess precise GPS coordinates that define the localization task. A query image is deemed localized when the GPS location of the highest-ranked database photo is within a 25-meter radius of the query image's position.

Table 3.1. Categorize data sets based on environmental factors

<table>
<tr><th colspan="3">Environments</th></tr>
<tr><th>World</th><th>Urban</th><th>Nature</th></tr>
<tr><td>YFCC 100M data set (Thomee et al., n.d.)</td><td>Google Street View Dataset (Amir Roshan Zamir & Shah, 2014)</td><td rowspan="4">Alps100K dataset (Cadík & Vaší˘, 2015)</td></tr>
<tr><td>IM2GPS dataset (Hays & Efros, 2008)</td><td>WorldCities dataset (Zemene et al., 2019)</td></tr>
<tr><td rowspan="2">VPRiCE dataset (“The vprice challenge 2015 visual place recognition in changing environments,” n.d.)</td><td>SanFrancisco data set (D. M. Chen et al., 2011)</td></tr>
<tr><td>Pittsburgh dataset (Torii & Pajdla, 2013)</td></tr>
</table>

3.6. Conclusion

In this chapter, we provided the basics of image based-localization field, as well as an overview of the prominent works in this area by reviewing the existing approaches based on image retrieval and classification. Then, we overviewed previous approaches based on structure from motion and data of multiple modalities. Finally, we discussed the literature related to deep learning as these approaches have been shown to enhance location recognition performance and pave the way for new computer vision applications that rely on comprehensive image content representations. Furthermore, we cite the most used datasets in these applications.

The next chapter will detail our contribution to improving the classification of Street View Image, followed by our validation results presentation.

CHAPTER 4

IMPROVING GOOGLE STREET VIEW IMAGE CLASSIFICATION USING FEATURES EXTRACTED FROM DIFFERENT CNN MODELS

CHAPTER 4: IMPROVING GOOGLE STREET VIEW IMAGE CLASSIFICATION USING FEATURES EXTRACTED FROM DIFFERENT CNN MODELS

4.1. Introduction

In prior chapters, we have presented an overview of image analysis and different artificial intelligence techniques to solve image geo-localization problem. We have explained in the first chapter the basics concepts of computer vision and everything related to digital images, especially its processing methods in feature extraction and classification phases. We also discussed the difficulties in computer vision. In the next one, we discussed the basics of artificial intelligence, machine learning and deep learning. Moreover, we defined the relation between them. We reviewed types of deep networks and their different components, architectures and applications. Then, in the third chapter, we presented a literature review on image-based geo-localization approaches by providing related works, different methods and available datasets to handle this task.

In this chapter, we will present our contribution to Street View Image classification and the obtained results. We will begin by outlining the overall architecture of proposed system and then study each of its two main phases separately. Finally, we will present our experimental design, analyze the results, and interpret their significance.

The results described in this chapter, about utilizing AlexNet Pre-trained CNN model were published in the international journal, "Periodica Polytechnica Electrical Engineering and Computer Science" (Djouadi & Kholladi, 2022).

4.2. Proposed approach

In this section, we outline a method for implementing a supervised learning model to classify Street View images by utilizing features extracted from various CNN models, which are AlexNet, VGG16, VGG19 and GoogleNet. Our method comprises of two phases; extraction features and classification. The primary objective is extracting features from input images utilizing several CNN models. Subsequently, the extracted features are used to classify the input image with the assistance of a SVM Classifier. This classification aids in determining Google Street View images location. The overview of our proposed approach is shown in Figure 4. 1.

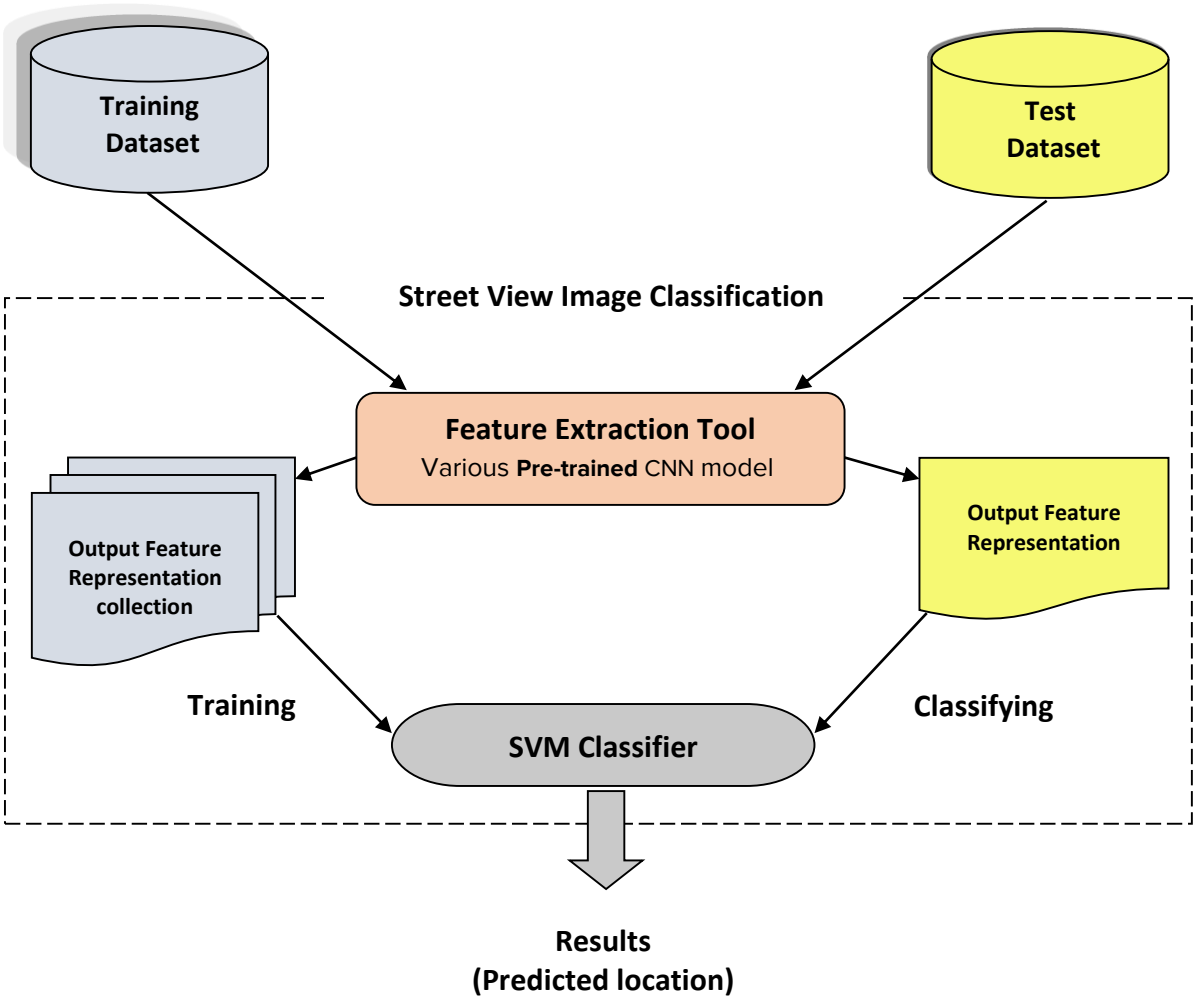

Figure 4.1. Flowchart of the proposed method for Google Street View image classification

4.2.1. Feature extraction phase

The feature extraction phase is the process of transforming the input image pixels to appropriate feature vectors. This phase is an essential and crucial step in our architecture because the Street View image classification performance largely depends on the choice of feature vectors used and on the quality of the feature extraction techniques used. Google Street View images are very complex and rich in their features; the feature extraction method plays an important role in improving the performance of Street View image classification. More precisely, the classification performance mainly depends on the power of the feature extraction methods.

We employed different pre-trained CNN architectures (AlexNet, VGG16, VGG19 and GoogleNet) to extract features, considering the fully connected layer output as representation of feature. The final fully connected layer was removed and characteristics were extracted from the second one, resulting in features vector of size 4096 from AlexNet, VGG16, VGG19, and 1024-sized feature vector from GoogleNet. The Figure. 4.2 illustrates the vector of feature obtained from the (FC2) of AlexNet model for direct image location prediction.

- **AlexNet architecture**

AlexNet model represets the convolutional neural network structure that developed by Alex Krizhevsky, Geoffrey E. Hinton and Ilya Sutskever (Krizhevsky et al., 2012). This very deep convolutional neural network includes additional filters per layer and accumulated convolutional layers shown in Figure. 4.3, consisting of five (5) convolutional layers, three (3) max-pooling layers, and three (3) fully connected layers. ReLU was utilized to reduce the time of training in AlexNet architecture, while dropout layers were employed to mitigate overfitting issues during training. The key finding of the original research study (Krizhevsky et al., 2012) highlighted that model depth significantly impacted its high performance; this was computationally intensive but made feasible through the utilization of Graphics Processing Units during training.

Table 4.1 illustrates the detailed architecture in Matlab. All images are resized to 227×227 pixels. In addition, AlexNet gives a 4096-dimensional feature vector from FC1 and FC2.

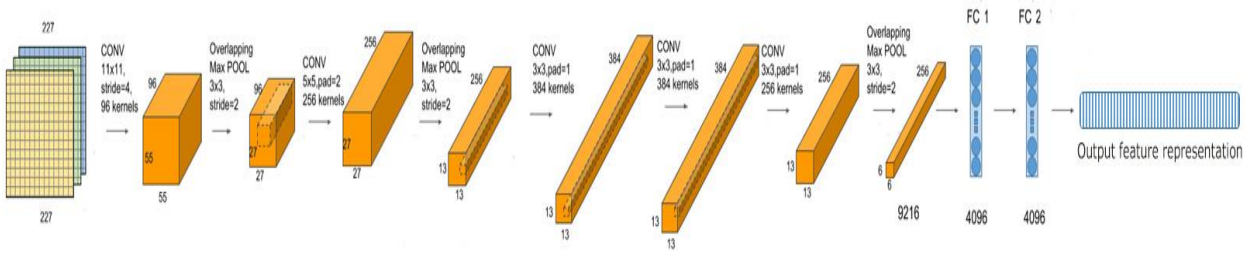

Figure 4.2 Vector of feature obtained from the second Fully Connected layer, its dimension is 4096

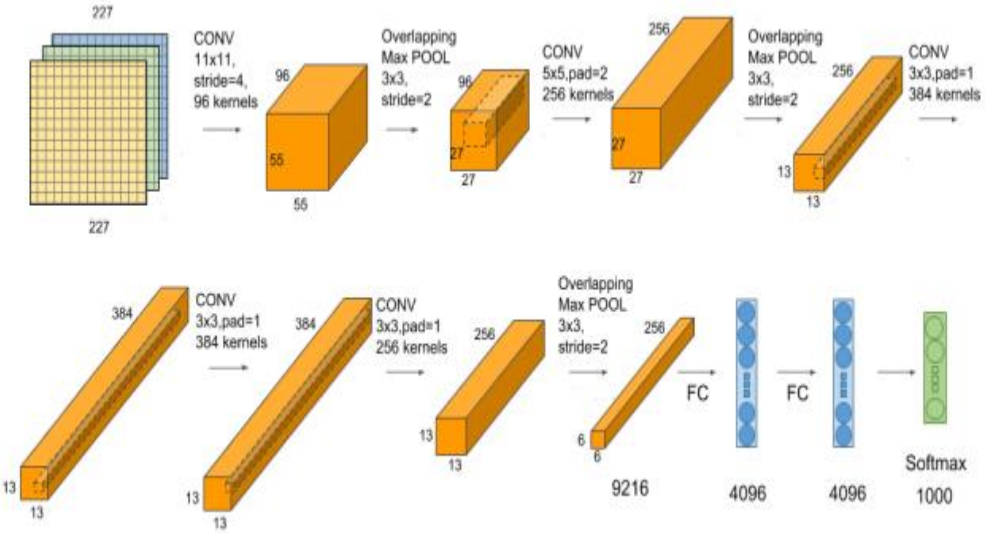

Figure 4.3. AlexNet architecture (designed and adapted from (Prijono, n.d.))

Table 4.1. Detailed architecture of AlexNet in Matlab

1	'data'	Image Input	227x227x3 images with 'zerocenter' normalization
2	'conv1'	Convolution	96 11x11x3 convolutions with stride [4 4] and padding [0 0]
3	'relu1'	ReLU	ReLU
4	'norm1'	Cross Channel Normalization	cross channel normalization with 5 channels per element
5	'pool1'	Max Pooling	3x3 max pooling with stride [2 2] and padding [0 0]
6	'conv2'	Convolution	256 5x5x48 convolutions with stride [1 1] and padding [2 2]
7	'relu2'	ReLU	ReLU
8	'norm2'	Cross Channel Normalization	cross channel normalization with 5 channels per element
9	'pool2'	Max Pooling	3x3 max pooling with stride [2 2] and padding [0 0]
10	'conv3'	Convolution	384 3x3x256 convolutions with stride [1 1] and padding [1 1]
11	'relu3'	ReLU	ReLU
12	'conv4'	Convolution	384 3x3x192 convolutions with stride [1 1] and padding [1 1]
13	'relu4'	ReLU	ReLU

14	'conv5'	Convolution	256 3x3x192 convolutions with stride [1 1] and padding [1 1]
15	'relu5'	ReLU	ReLU
16	'pool5'	Max Pooling	3x3 max pooling with stride [2 2] and padding [0 0]
17	'fc6'	Fully Connected	4096 fully connected layer
18	'relu6'	ReLU	ReLU
19	'drop6'	Dropout	50% dropout
20	'fc7'	Fully Connected	4096 fully connected layer
21	'relu7'	ReLU	ReLU
22	'drop7'	Dropout	50% dropout
23	'fc8'	Fully Connected	1000 fully connected layer
24	'prob'	Softmax	softmax
25	'output'	Classification Output	crossentropyex with 'tench', 'goldfish', and 998 other classes

- **VGGNet architecture**

The impressive results of the two very deep CNNs, VGG-16 and VGG-19, demonstrate the important role that network depth plays in improving classification performance. The depth of convolution forces the network to learn more abstract and discriminative representations. Deeper layers result in more complex learned features, which is why we utilized both VGG-16 and VGG-19 in our work. VGG-16 (containing thirteen (13) convolutional layers and three (3) fully connected layers) and VGG-19 (containing sixteen (16) convolutional layers and three (3) fully connected layers).

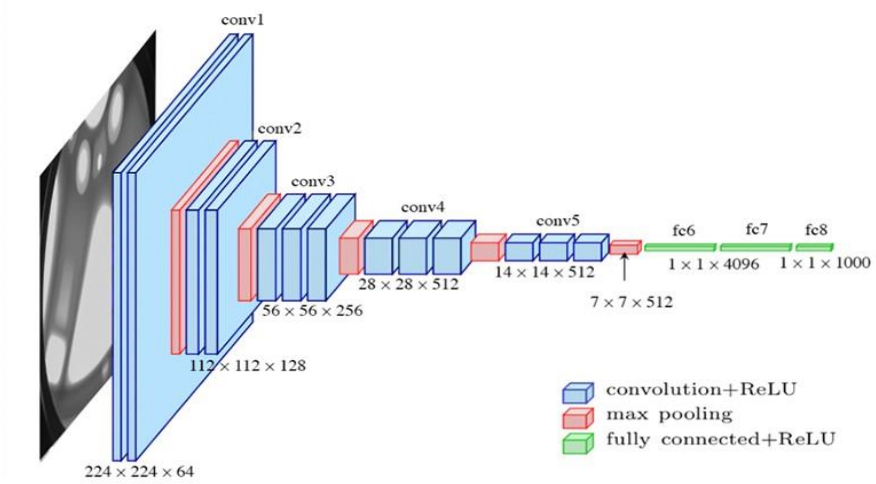

Figure 4.4. Architecture of VGGNet

Tables 4.2 and 4.3 respectively present the detailed architecture of VGG-16 and VGG-19 from Matlab. Furthermore, all images are resized to 224×224, and VGG-16 and VGG-19 generate feature vectors of 4096-dimensions from FC1 or FC2.

- **Google Net architecture**

GoogleNet was among the first models to propose that CNN layers do not always have to be stacked sequentially. The introduction of the Inception module allowed for a creative structuring of the layers, leading to improved classification performance and computational efficiency. We evaluated their performance for Street View image classification in our work due to their architecture and depth.

Table 4.4 details the architecture of GoogleNet from Matlab. All samples are resized to 224×224 pixels. GoogLeNet provides 1024-dimensional feature vectors from the last pooled layer or drop pooled layer.

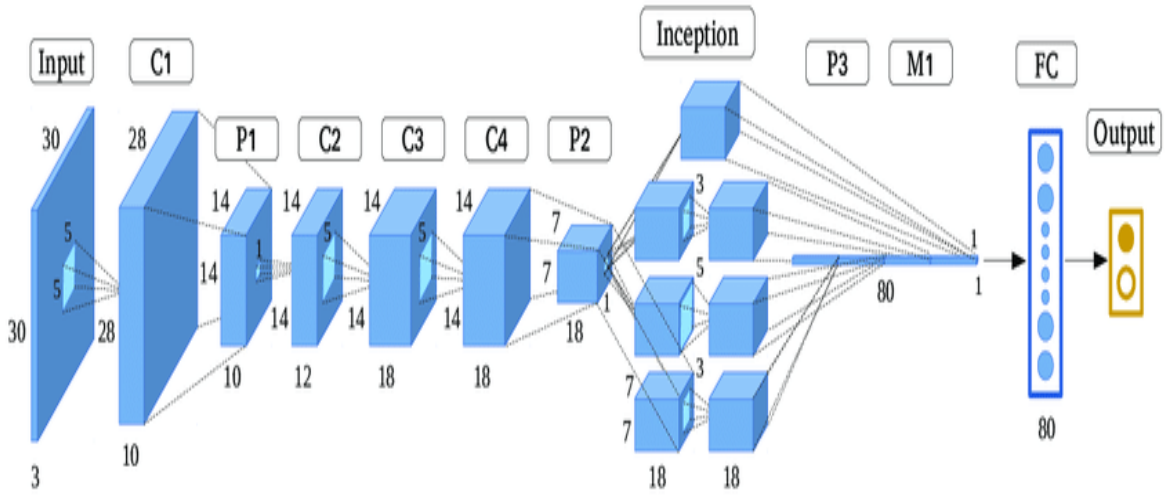

Figure 4.5. Architecture of GoogleNet

Table 4.2. Detailed architecture of VGG16 in Matlab

41x1 Layer array with layers:

1	'input'	Image Input	224x224x3 images with 'zerocenter' normalization
2	'conv1_1'	Convolution	64 3x3x3 convolutions with stride [1 1] and padding [1 1]
3	'relu1_1'	ReLU	ReLU
4	'conv1_2'	Convolution	64 3x3x64 convolutions with stride [1 1] and padding [1 1]
5	'relu1_2'	ReLU	ReLU
6	'pool1'	Max Pooling	2x2 max pooling with stride [2 2] and padding [0 0]
7	'conv2_1'	Convolution	128 3x3x64 convolutions with stride [1 1] and padding [1 1]
8	'relu2_1'	ReLU	ReLU
9	'conv2_2'	Convolution	128 3x3x128 convolutions with stride [1 1] and padding [1 1]
10	'relu2_2'	ReLU	ReLU
11	'pool2'	Max Pooling	2x2 max pooling with stride [2 2] and padding [0 0]
12	'conv3_1'	Convolution	256 3x3x128 convolutions with stride [1 1] and padding [1 1]
13	'relu3_1'	ReLU	ReLU
14	'conv3_2'	Convolution	256 3x3x256 convolutions with stride [1 1] and padding [1 1]
15	'relu3_2'	ReLU	ReLU
16	'conv3_3'	Convolution	256 3x3x256 convolutions with stride [1 1] and padding [1 1]
17	'relu3_3'	ReLU	ReLU
18	'pool3'	Max Pooling	2x2 max pooling with stride [2 2] and padding [0 0]
19	'conv4_1'	Convolution	512 3x3x256 convolutions with stride [1 1] and padding [1 1]
20	'relu4_1'	ReLU	ReLU
21	'conv4_2'	Convolution	512 3x3x512 convolutions with stride [1 1] and padding [1 1]

22	'relu4_2'	ReLU	ReLU
23	'conv4_3'	Convolution	512 3x3x512 convolutions with stride [1 1] and padding [1 1]
24	'relu4_3'	ReLU	ReLU
25	'pool4'	Max Pooling	2x2 max pooling with stride [2 2] and padding [0 0]
26	'conv5_1'	Convolution	512 3x3x512 convolutions with stride [1 1] and padding [1 1]
27	'relu5_1'	ReLU	ReLU
28	'conv5_2'	Convolution	512 3x3x512 convolutions with stride [1 1] and padding [1 1]
29	'relu5_2'	ReLU	ReLU
30	'conv5_3'	Convolution	512 3x3x512 convolutions with stride [1 1] and padding [1 1]
31	'relu5_3'	ReLU	ReLU
32	'pool5'	Max Pooling	2x2 max pooling with stride [2 2] and padding [0 0]
33	'fc6'	Fully Connected	4096 fully connected layer
34	'relu6'	ReLU	ReLU
35	'drop6'	Dropout	50% dropout
36	'fc7'	Fully Connected	4096 fully connected layer
37	'relu7'	ReLU	ReLU
38	'drop7'	Dropout	50% dropout
39	'fc8'	Fully Connected	1000 fully connected layer
40	'prob'	Softmax	softmax
41	'output'	Classification Output	crossentropyex with 'tench', 'goldfish', and 998 other classes

Table 4.3. Detailed architecture of VGG19 in Matlab

47x1 Layer array with layers:

1	'input'	Image Input	224x224x3 images with 'zerocenter' normalization
2	'conv1_1'	Convolution	64 3x3x3 convolutions with stride [1 1] and padding [1 1]
3	'relu1_1'	ReLU	ReLU
4	'conv1_2'	Convolution	64 3x3x64 convolutions with stride [1 1] and padding [1 1]
5	'relu1_2'	ReLU	ReLU
6	'pool1'	Max Pooling	2x2 max pooling with stride [2 2] and padding [0 0]
7	'conv2_1'	Convolution	128 3x3x64 convolutions with stride [1 1] and padding [1 1]
8	'relu2_1'	ReLU	ReLU
9	'conv2_2'	Convolution	128 3x3x128 convolutions with stride [1 1] and padding [1 1]
10	'relu2_2'	ReLU	ReLU
11	'pool2'	Max Pooling	2x2 max pooling with stride [2 2] and padding [0 0]
12	'conv3_1'	Convolution	256 3x3x128 convolutions with stride [1 1] and padding [1 1]
13	'relu3_1'	ReLU	ReLU
14	'conv3_2'	Convolution	256 3x3x256 convolutions with stride [1 1] and padding [1 1]
15	'relu3_2'	ReLU	ReLU
16	'conv3_3'	Convolution	256 3x3x256 convolutions with stride [1 1] and padding [1 1]
17	'relu3_3'	ReLU	ReLU
18	'conv3_4'	Convolution	256 3x3x256 convolutions with stride [1 1] and padding [1 1]
19	'relu3_4'	ReLU	ReLU
20	'pool3'	Max Pooling	2x2 max pooling with stride [2 2] and padding [0 0]
21	'conv4_1'	Convolution	512 3x3x256 convolutions with stride [1 1] and padding [1 1]
22	'relu4_1'	ReLU	ReLU

23	'conv4_2'	Convolution	512 3x3x512 convolutions with stride [1 1] and padding [1 1]
24	'relu4_2'	ReLU	ReLU
25	'conv4_3'	Convolution	512 3x3x512 convolutions with stride [1 1] and padding [1 1]
26	'relu4_3'	ReLU	ReLU
27	'conv4_4'	Convolution	512 3x3x512 convolutions with stride [1 1] and padding [1 1]
28	'relu4_4'	ReLU	ReLU
29	'pool4'	Max Pooling	2x2 max pooling with stride [2 2] and padding [0 0]
30	'conv5_1'	Convolution	512 3x3x512 convolutions with stride [1 1] and padding [1 1]
31	'relu5_1'	ReLU	ReLU
32	'conv5_2'	Convolution	512 3x3x512 convolutions with stride [1 1] and padding [1 1]
33	'relu5_2'	ReLU	ReLU
34	'conv5_3'	Convolution	512 3x3x512 convolutions with stride [1 1] and padding [1 1]
35	'relu5_3'	ReLU	ReLU
36	'conv5_4'	Convolution	512 3x3x512 convolutions with stride [1 1] and padding [1 1]
37	'relu5_4'	ReLU	ReLU
38	'pool5'	Max Pooling	2x2 max pooling with stride [2 2] and padding [0 0]
39	'fc6'	Fully Connected	4096 fully connected layer
40	'relu6'	ReLU	ReLU
41	'drop6'	Dropout	50% dropout
42	'fc7'	Fully Connected	4096 fully connected layer
43	'relu7'	ReLU	ReLU
44	'drop7'	Dropout	50% dropout
45	'fc8'	Fully Connected	1000 fully connected layer
46	'prob'	Softmax	softmax
47	'output'	Classification Output	crossentropyex with 'tench', 'goldfish', and 998 other classes

Table 4.4. Detailed architecture of GoogleNet in Matlab

144x1 Layer array with layers:

1	'data'	Image Input	224x224x3 images with 'zerocenter' normalization
2	'conv1-7x7_s2'	Convolution	64 7x7x3 convolutions with stride [2 2] and padding [3 3 3 3]
3	'conv1-relu_7x7'	ReLU	ReLU
4	'pool1-3x3_s2'	Max Pooling	3x3 max pooling with stride [2 2] and padding [0 1 0 1]
5	'pool1-norm1'	Cross Channel Normalization	cross channel normalization with 5 channels per element
6	'conv2-3x3_reduce'	Convolution	64 1x1x64 convolutions with stride [1 1] and padding [0 0 0 0]
7	'conv2-relu_3x3_reduce'	ReLU	ReLU
8	'conv2-3x3'	Convolution	192 3x3x64 convolutions with stride [1 1] and padding [1 1 1 1]
9	'conv2-relu_3x3'	ReLU	ReLU
10	'conv2-norm2'	Cross Channel Normalization	cross channel normalization with 5 channels per element
11	'pool2-3x3_s2'	Max Pooling	3x3 max pooling with stride [2 2] and padding [0 1 0 1]
12	'inception_3a-1x1'	Convolution	64 1x1x192 convolutions with stride [1 1] and padding [0 0 0 0]
13	'inception_3a-relu_1x1'	ReLU	ReLU
14	'inception_3a-3x3_reduce'	Convolution	96 1x1x192 convolutions with stride [1 1] and padding [0 0 0 0]
15	'inception_3a-relu_3x3_reduce'	ReLU	ReLU
16	'inception_3a-3x3'	Convolution	128 3x3x96 convolutions with stride [1 1] and padding [1 1 1 1]
17	'inception_3a-relu_3x3'	ReLU	ReLU
18	'inception_3a-5x5_reduce'	Convolution	16 1x1x192 convolutions with stride [1 1] and padding [0 0 0 0]
19	'inception_3a-relu_5x5_reduce'	ReLU	ReLU

20	'inception_3a-5x5'	Convolution	32 5x5x16 convolutions with stride [1 1] and padding [2 2 2 2]
21	'inception_3a-relu_5x5'	ReLU	ReLU
22	'inception_3a-pool'	Max Pooling	3x3 max pooling with stride [1 1] and padding [1 1 1 1]
23	'inception_3a-pool_proj'	Convolution	32 1x1x192 convolutions with stride [1 1] and padding [0 0 0 0]
24	'inception_3a-relu_pool_proj'	ReLU	ReLU
25	'inception_3a-output'	Depth concatenation	Depth concatenation of 4 inputs
26	'inception_3b-1x1'	Convolution	128 1x1x256 convolutions with stride [1 1] and padding [0 0 0 0]
27	'inception_3b-relu_1x1'	ReLU	ReLU
28	'inception_3b-3x3_reduce'	Convolution	128 1x1x256 convolutions with stride [1 1] and padding [0 0 0 0]
29	'inception_3b-relu_3x3_reduce'	ReLU	ReLU
30	'inception_3b-3x3'	Convolution	192 3x3x128 convolutions with stride [1 1] and padding [1 1 1 1]
31	'inception_3b-relu_3x3'	ReLU	ReLU
32	'inception_3b-5x5_reduce'	Convolution	32 1x1x256 convolutions with stride [1 1] and padding [0 0 0 0]
33	'inception_3b-relu_5x5_reduce'	ReLU	ReLU
34	'inception_3b-5x5'	Convolution	96 5x5x32 convolutions with stride [1 1] and padding [2 2 2 2]
35	'inception_3b-relu_5x5'	ReLU	ReLU
36	'inception_3b-pool'	Max Pooling	3x3 max pooling with stride [1 1] and padding [1 1 1 1]
37	'inception_3b-pool_proj'	Convolution	64 1x1x256 convolutions with stride [1 1] and padding [0 0 0 0]
38	'inception_3b-relu_pool_proj'	ReLU	ReLU
39	'inception_3b-output'	Depth concatenation	Depth concatenation of 4 inputs
40	'pool3-3x3_s2'	Max Pooling	3x3 max pooling with stride [2 2] and padding [0 1 0 1]
41	'inception_4a-1x1'	Convolution	192 1x1x480 convolutions with stride [1 1] and padding [0 0 0 0]

42	'inception_4a-relu_1x1'	ReLU	ReLU
43	'inception_4a-3x3_reduce'	Convolution	96 1x1x480 convolutions with stride [1 1] and padding [0 0 0 0]
44	'inception_4a-relu_3x3_reduce'	ReLU	ReLU
45	'inception_4a-3x3'	Convolution	208 3x3x96 convolutions with stride [1 1] and padding [1 1 1 1]
46	'inception_4a-relu_3x3'	ReLU	ReLU
47	'inception_4a-5x5_reduce'	Convolution	16 1x1x480 convolutions with stride [1 1] and padding [0 0 0 0]
48	'inception_4a-relu_5x5_reduce'	ReLU	ReLU
49	'inception_4a-5x5'	Convolution	48 5x5x16 convolutions with stride [1 1] and padding [2 2 2 2]
50	'inception_4a-relu_5x5'	ReLU	ReLU
51	'inception_4a-pool'	Max Pooling	3x3 max pooling with stride [1 1] and padding [1 1 1 1]
52	'inception_4a-pool_proj'	Convolution	64 1x1x480 convolutions with stride [1 1] and padding [0 0 0 0]
53	'inception_4a-relu_pool_proj'	ReLU	ReLU
54	'inception_4a-output'	Depth concatenation	Depth concatenation of 4 inputs
55	'inception_4b-1x1'	Convolution	160 1x1x512 convolutions with stride [1 1] and padding [0 0 0 0]
56	'inception_4b-relu_1x1'	ReLU	ReLU
57	'inception_4b-3x3_reduce'	Convolution	112 1x1x512 convolutions with stride [1 1] and padding [0 0 0 0]
58	'inception_4b-relu_3x3_reduce'	ReLU	ReLU
59	'inception_4b-3x3'	Convolution	224 3x3x112 convolutions with stride [1 1] and padding [1 1 1 1]
60	'inception_4b-relu_3x3'	ReLU	ReLU
61	'inception_4b-5x5_reduce'	Convolution	24 1x1x512 convolutions with stride [1 1] and padding [0 0 0 0]
62	'inception_4b-relu_5x5_reduce'	ReLU	ReLU
63	'inception_4b-5x5'	Convolution	64 5x5x24 convolutions with stride [1 1] and padding [2 2 2 2]

64	'inception_4b-relu_5x5'	ReLU	ReLU
65	'inception_4b-pool'	Max Pooling	3x3 max pooling with stride [1 1] and padding [1 1 1 1]
66	'inception_4b-pool_proj'	Convolution	64 1x1x512 convolutions with stride [1 1] and padding [0 0 0 0]
67	'inception_4b-relu_pool_proj'	ReLU	ReLU
68	'inception_4b-output'	Depth concatenation	Depth concatenation of 4 inputs
69	'inception_4c-1x1'	Convolution	128 1x1x512 convolutions with stride [1 1] and padding [0 0 0 0]
70	'inception_4c-relu_1x1'	ReLU	ReLU
71	'inception_4c-3x3_reduce'	Convolution	128 1x1x512 convolutions with stride [1 1] and padding [0 0 0 0]
72	'inception_4c-relu_3x3_reduce'	ReLU	ReLU
73	'inception_4c-3x3'	Convolution	256 3x3x128 convolutions with stride [1 1] and padding [1 1 1 1]
74	'inception_4c-relu_3x3'	ReLU	ReLU
75	'inception_4c-5x5_reduce'	Convolution	24 1x1x512 convolutions with stride [1 1] and padding [0 0 0 0]
76	'inception_4c-relu_5x5_reduce'	ReLU	ReLU
77	'inception_4c-5x5'	Convolution	64 5x5x24 convolutions with stride [1 1] and padding [2 2 2 2]
78	'inception_4c-relu_5x5'	ReLU	ReLU
79	'inception_4c-pool'	Max Pooling	3x3 max pooling with stride [1 1] and padding [1 1 1 1]
80	'inception_4c-pool_proj'	Convolution	64 1x1x512 convolutions with stride [1 1] and padding [0 0 0 0]
81	'inception_4c-relu_pool_proj'	ReLU	ReLU
82	'inception_4c-output'	Depth concatenation	Depth concatenation of 4 inputs
83	'inception_4d-1x1'	Convolution	112 1x1x512 convolutions with stride [1 1] and padding [0 0 0 0]
84	'inception_4d-relu_1x1'	ReLU	ReLU
85	'inception_4d-3x3_reduce'	Convolution	144 1x1x512 convolutions with stride [1 1] and padding [0 0 0 0]

86	'inception_4d-relu_3x3_reduce'	ReLU	ReLU
87	'inception_4d-3x3'	Convolution	288 3x3x144 convolutions with stride [1 1] and padding [1 1 1 1]
88	'inception_4d-relu_3x3'	ReLU	ReLU
89	'inception_4d-5x5_reduce'	Convolution	32 1x1x512 convolutions with stride [1 1] and padding [0 0 0 0]
90	'inception_4d-relu_5x5_reduce'	ReLU	ReLU
91	'inception_4d-5x5'	Convolution	64 5x5x32 convolutions with stride [1 1] and padding [2 2 2 2]
92	'inception_4d-relu_5x5'	ReLU	ReLU
93	'inception_4d-pool'	Max Pooling	3x3 max pooling with stride [1 1] and padding [1 1 1 1]
94	'inception_4d-pool_proj'	Convolution	64 1x1x512 convolutions with stride [1 1] and padding [0 0 0 0]
95	'inception_4d-relu_pool_proj'	ReLU	ReLU
96	'inception_4d-output'	Depth concatenation	Depth concatenation of 4 inputs
97	'inception_4e-1x1'	Convolution	256 1x1x528 convolutions with stride [1 1] and padding [0 0 0 0]
98	'inception_4e-relu_1x1'	ReLU	ReLU
99	'inception_4e-3x3_reduce'	Convolution	160 1x1x528 convolutions with stride [1 1] and padding [0 0 0 0]
100	'inception_4e-relu_3x3_reduce'	ReLU	ReLU
101	'inception_4e-3x3'	Convolution	320 3x3x160 convolutions with stride [1 1] and padding [1 1 1 1]
102	'inception_4e-relu_3x3'	ReLU	ReLU
103	'inception_4e-5x5_reduce'	Convolution	32 1x1x528 convolutions with stride [1 1] and padding [0 0 0 0]
104	'inception_4e-relu_5x5_reduce'	ReLU	ReLU
105	'inception_4e-5x5'	Convolution	128 5x5x32 convolutions with stride [1 1] and padding [2 2 2 2]
106	'inception_4e-relu_5x5'	ReLU	ReLU

107	'inception_4e-pool'	Max Pooling	3x3 max pooling with stride [1 1] and padding [1 1 1 1]
108	'inception_4e-pool_proj'	Convolution	128 1x1x528 convolutions with stride [1 1] and padding [0 0 0 0]
109	'inception_4e-relu_pool_proj'	ReLU	ReLU
110	'inception_4e-output'	Depth concatenation	Depth concatenation of 4 inputs
111	'pool4-3x3_s2'	Max Pooling	3x3 max pooling with stride [2 2] and padding [0 1 0 1]
112	'inception_5a-1x1'	Convolution	256 1x1x832 convolutions with stride [1 1] and padding [0 0 0 0]
113	'inception_5a-relu_1x1'	ReLU	ReLU
114	'inception_5a-3x3_reduce'	Convolution	160 1x1x832 convolutions with stride [1 1] and padding [0 0 0 0]
115	'inception_5a-relu_3x3_reduce'	ReLU	ReLU
116	'inception_5a-3x3'	Convolution	320 3x3x160 convolutions with stride [1 1] and padding [1 1 1 1]
117	'inception_5a-relu_3x3'	ReLU	ReLU
118	'inception_5a-5x5_reduce'	Convolution	32 1x1x832 convolutions with stride [1 1] and padding [0 0 0 0]
119	'inception_5a-relu_5x5_reduce'	ReLU	ReLU
120	'inception_5a-5x5'	Convolution	128 5x5x32 convolutions with stride [1 1] and padding [2 2 2 2]
121	'inception_5a-relu_5x5'	ReLU	ReLU
122	'inception_5a-pool'	Max Pooling	3x3 max pooling with stride [1 1] and padding [1 1 1 1]
123	'inception_5a-pool_proj'	Convolution	128 1x1x832 convolutions with stride [1 1] and padding [0 0 0 0]
124	'inception_5a-relu_pool_proj'	ReLU	ReLU

125	'inception_5a-output'	Depth concatenation	Depth concatenation of 4 inputs
126	'inception_5b-1x1'	Convolution	384 1x1x832 convolutions with stride [1 1] and padding [0 0 0]
127	'inception_5b-relu_1x1'	ReLU	ReLU
128	'inception_5b-3x3_reduce'	Convolution	192 1x1x832 convolutions with stride [1 1] and padding [0 0 0 0]
129	'inception_5b-relu_3x3_reduce'	ReLU	ReLU
130	'inception_5b-3x3'	Convolution	384 3x3x192 convolutions with stride [1 1] and padding [1 1 1 1]
131	'inception_5b-relu_3x3'	ReLU	ReLU
132	'inception_5b-5x5_reduce'	Convolution	48 1x1x832 convolutions with stride [1 1] and padding [0 0 0 0]
133	'inception_5b-relu_5x5_reduce'	ReLU	ReLU
134	'inception_5b-5x5'	Convolution	128 5x5x48 convolutions with stride [1 1] and padding [2 2 2 2]
135	'inception_5b-relu_5x5'	ReLU	ReLU
136	'inception_5b-pool'	Max Pooling	3x3 max pooling with stride [1 1] and padding [1 1 1 1]
137	'inception_5b-pool_proj'	Convolution	128 1x1x832 convolutions with stride [1 1] and padding [0 0 0 0]
138	'inception_5b-relu_pool_proj'	ReLU	ReLU
139	'inception_5b-output'	Depth concatenation	Depth concatenation of 4 inputs
140	'pool5-7x7_s1'	Average Pooling	7x7 average pooling with stride [1 1] and padding [0 0 0 0]
141	'pool5-drop_7x7_s1'	Dropout	40% dropout
142	'loss3-classifier'	Fully Connected	1000 fully connected layer
143	'prob'	Softmax	softmax
144	'output'	Classification Output	crossentropyex with 'tench', 'goldfish', and 998 other classes>>

4.2.2. Google Street View image classification phase

In the previous step, we employed a supervised learning method that relied on deep feature descriptors to detect the position of input images through classification. We choose to utilize the Support Vector Machine because it is adept at handling high-dimensional input vectors, enabling accurate classification and prediction of image locations. This capability makes it an important tool for diverse applications including geolocation-based services and precise image recognition systems. Furthermore, our use of SVM for image classification and prediction in this context showcases its effectiveness in handling complex datasets and achieving high accuracy levels. Additionally, the versatility of SVM extends across various domains due to its robustness with multidimensional data representations.

- **Training**

During training phase, SVM takes the collection of output feature representation along with labels of their input images to produce a decision function that maximizes the margin between classes. This involves initializing the base of the models by saving the characteristics of different classes with their corresponding labels, as illustrated in Figure 4.6.

- **Test**

This phase consists of assigning a class for each new example given based on the data in the model database. More precisely, in this phase, SVM only takes output feature representation without any labels and compares it with the existing models in the database of models obtained from the training phase to make a classification decision as illustrated in Figure 4.6.

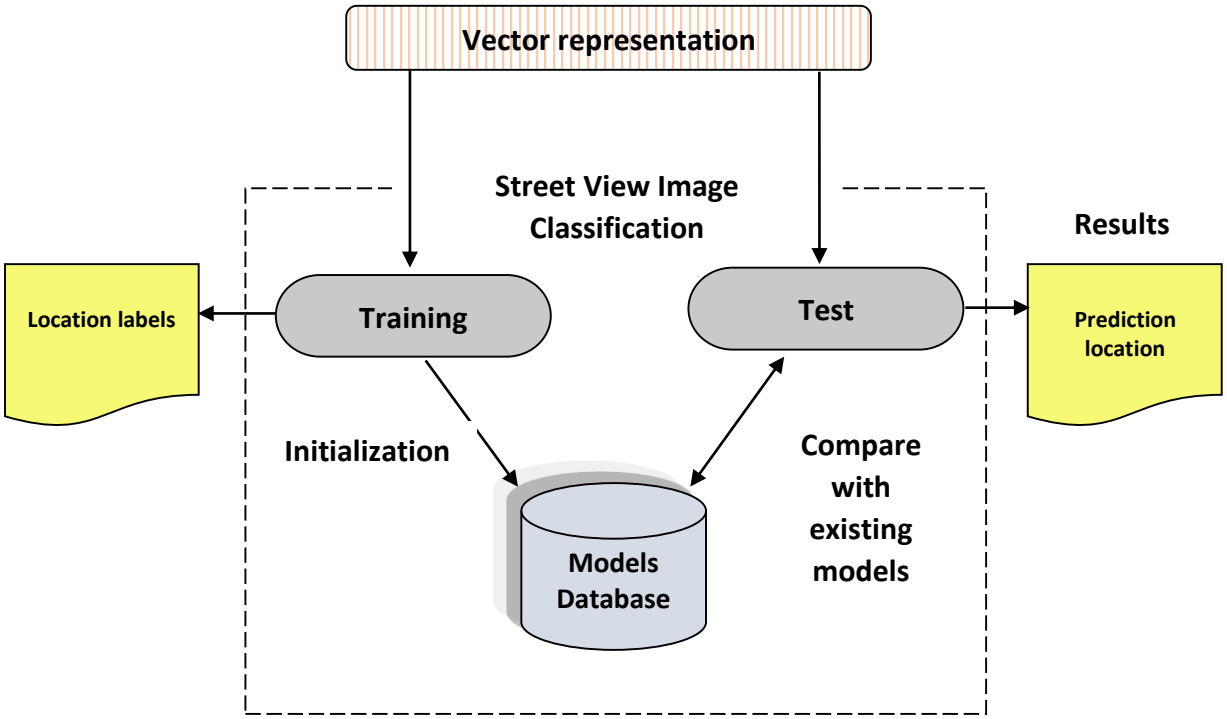

Figure 4.6. Classification phase

4.3. Experimental results

In this part, we assess the effectiveness of the proposed method for determining the geographical location of Google Street View photos captured in urban areas, which typically contain diverse visual cues revealing their locations. The dataset, thorough experiments, and resulting findings are thoroughly explained.

4.3.1. Evaluation dataset

We evaluate the performance of the proposed approach using the dataset of Google Street View (Amir Roshan Zamir & Shah, 2014), which includes 62 058 high-resolution imagery. These images display the downtown area and nearby regions of Pittsburgh, PA; Orlando, FL and Manhattan, NY. The 360-degree view of each location marker is splied into four (4) side views and one (1) top-view image. For each location there is an additional image displaying information like addresses and names of streets. The Google Street View dataset is a valuable resource to evaluate the effectiveness of our algorithm, as it contains a substantial number of high-quality images captured in different locations and under varying conditions. Figure 4.7 provides a sample of street view images from six (6) place marks of the dataset.

Figure 4.7. Sample of street view images

4.3.2. Experiment setup

All experiments were performed on an HP ProLiant DL380 G7 Server operating on Windows Server 2012 R2 Essentials, equipped with a 6-core Intel Xeon CPU E 5649 running at a frequency of 2.53 GHz, while the system is equipped with 32 GB of RAM. For the execution of our project's program, we employed Matlab version R 2017b (9.3.0. 713579) 64- bit, which is software that enables efficient solutions to intricate numerical problems within minimal time and provides convenient image manipulation features.

Matlab was designed to make matrix computations easy, and its functionality can be significantly extended by adding toolboxes. These toolboxes are folders that contain a variety of built-in functions and tested tools, grouped according to their specific use.

The base CNN model utilized in this research are AlexNet, VGG16, VGG19 and GoogleNet can be accessed through Matlab/Simulink. It is called from the following instructions:

```
netA = alexnet;

netB = vgg16;

netC = vgg19;

netD = googlenet;
```

All samples are resized using the *imresize* function in Matlab/Simulink before feeding the feature vector extracted into a linear SVM classifier.

```
I1 = imresize(I,[227,227]); ## for alexnet

I2 = imresize(I,[224,224]); ## for vgg16,vgg19,googlenet
```

4.3.3. Performance measurement and statistical validation

In our experiments, we randomly divided the dataset into 90% training samples and 10% testing samples. We conducted a total of 60 runs and then averaged the results to ensure statistical significance. For each run, the classification performance was evaluated using the accuracy (%) for assessing the models classifying as a metric. It calculated by taking the correct predictions number and dividing it by the total predictions number, also it could be calculated in terms of positives and negatives outcomes as demonstrated in the following equation:

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (4.1)$$

Where TP = True Positives, TN = True Negatives,
FP = False Positives, and FN = False Negatives.

To get accurate inferences from the numerical results, we conducted a statistical study by using Graph Pad Prism (version9.2.0 for Windows) on our outcomes. This software program is a statistics, data analysis and graphing package that simplifies powerful statistics to help users analyze smarter, not harder.

To achieve this, we initially employed the test of Kolmogorov-Smirnov with the Dallal-Wilkinson-Lilliefors p-value to ascertain whether the dataset follows a normal distribution. Subsequently, we utilized the D'Agostino-Pearson test due to its flexibility and strong recommendation as a normality test. This testing method generally requires a sample size of 20 or more and is advisable for datasets containing repeated values. For all statistical analyses, significance was determined at $p < 0.05$.

GENERAL CONCLUSION AND FUTURE PROSPECTS

GENERAL CONCLUSION AND FUTURE PROSPECTS

The problems of geo-localization and place recognition have been extensively studied, leading to the development of numerous solutions. This thesis provides an overview of recent methodologies and approaches employed in addressing these challenges. Initially, large-scale geo-localization has been explored using global localization methods that leverage machine learning techniques and extensive image databases. Despite the abundance of available images, achieving accurate global geo-localization for practical purposes still remains a challenge.

Recently, the majority of research has been focused on city-scale geo-localization, particularly utilizing image retrieval methods in large urban environments. However, a key challenge lies in conducting efficient searches within extensive databases for this type of geo-localization research. To address these challenges, innovative approaches involving Deep Learning techniques have been introduced. A fundamental discovery is that these methods can effectively learn to adapt to changes in scene appearance such as variations in illumination (daytime vs nighttime) and seasonal transformations (winter vs summer) by harnessing the Google Time Machine feature of street view imagery; thus enhancing advancements across various issues within this field.

Our focus is on solving the issue of image geo-localization through the utilization of pre-trained CNN model extracted features and the Support Vector Machine classifier to enhance Street View image classification performance. We achieved our goal of identifying the location from images by combining Pre-trained CNN models for features extraction with the SVM classifier. To evaluate the effectiveness of this approach, we utilized the high-quality Google Street View dataset encompassing downtown and surrounding areas of

Pittsburgh, PA; Orlando, FL; and Manhattan, NY. Our proposed method's efficiency was substantiated through statistical validation.

The results of the experiment indicate that the advanced attributes obtained from the second fully connected layer (FC2) of pre-trained CNN models such as AlexNet, VGG16, VGG19, and GoogleNet can greatly improve the classification accuracies.

In future projects, we aim to incorporate additional pre-trained Convolutional Neural Networks (CNN) models to improve the accuracy of classifying Street View image. Moreover, we intend to assess the effectiveness of our suggested approach using various existing datasets.

REFERENCES

Al-obaide, Z. H., & Al-ani, A. A. (2022). Comparison study between image retrieval methods. *Iraqi Journal of Information and Communication Technology(IJICT)*, *5*(1), 1–15. https://doi.org/10.31987/ijict.5.1.182

Alzubaidi, L., Zhang, J., Humaidi, A. J., Dujaili, A. Al, Duan, Y., Shamma, O. Al, … Farhan, L. (2021). Review of deep learning : concepts , CNN architectures , challenges , applications , future directions. *Journal of Big Data*, *8*(53), 1.74. https://doi.org/10.1186/s40537-021-00444-8

Alzubi, A., Amira, A., & Ramzan, N. (2015). Semantic Content-based Image Retrieval : A Comprehensive Study. *JOURNAL OF VISUAL COMMUNICATION AND IMAGE REPRESENTATION*, (July). https://doi.org/10.1016/j.jvcir.2015.07.012

Andrade, F. S. P., Almeida, J., Pedrini, H., & Torres, R. D. S. (2012). Fusion of local and global descriptors for content-based image and video retrieval. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *7441 LNCS*, 845–853. https://doi.org/10.1007/978-3-642-33275-3_104

Arandjelovi, R., Gronat, P., Torii, A., Pajdla, T., Sivic, J., Arandjelovi, R., … Pajdla, T. (2016). NetVLAD : CNN architecture for weakly supervised place recognition To cite this version :

Arandjelovic, R., Gronat, P., Torii, A., Pajdla, T., & Sivic, J. (2018). NetVLAD: CNN Architecture for Weakly Supervised Place Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *40*(6), 1437–1451. https://doi.org/10.1109/TPAMI.2017.2711011

Ashfaque, J. M., & Iqbal, A. (2019). Introduction to Support Vector Machines and Kernel Methods, (April), 1–10.

Aslam, M. A., Salik, M. N., Chughtai, F., Ali, N., Dar, S. H., & Khalil, T. (2019). Image classification based on mid-level feature fusion. *15th International Conference on Emerging Technologies, ICET 2019*. https://doi.org/10.1109/ICET48972.2019.8994721

Avrithis, Y., Kalantidis, Y., Tolias, G., & Spyrou, E. (2010). Retrieving Landmark and Non-Landmark Images from Community Photo Collections Categories and Subject Descriptors, 153–162.

Azzam, R., Taha, T., Huang, S., & Zweiri, Y. (2020). Feature‐based visual simultaneous localization and mapping: a survey. *SN Applied Sciences*, *2*(2), 1–24. https://doi.org/10.1007/s42452-020-2001-3

Baatz, G., Köser, K., Chen, D., Grzeszczuk, R., & Pollefeys, M. (2010). Handling urban location recognition as a 2D homothetic problem. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *6316 LNCS*(PART 6), 266–279. https://doi.org/10.1007/978-3-642-15567-3_20

Baatz, G., Saurer, O., Köser, K., & Pollefeys, M. (2012). Large scale visual geo-localization of images in mountainous terrain. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *7573 LNCS*(PART 2), 517–530. https://doi.org/10.1007/978-3-642-33709-3_37

Bansal, Mayank, Sawhney, H. S., Cheng, H., & Daniilidis, K. (2011). Geo-Localization of Street Views with Aerial Image Databases, 1125–1128.

Bansal, Monika, Kumar, M., Sachdeva, M., & Mittal, A. (2021). Transfer learning for image classification using VGG19: Caltech-101 image data set. *Journal of Ambient Intelligence and Humanized Computing*, (0123456789). https://doi.org/10.1007/s12652-021-03488-z

Bergamo, A., & Torresani, L. (2013). Leveraging Structure from Motion to Learn Discriminative Codebooks for Scalable Landmark Classification. https://doi.org/10.1109/CVPR.2013.104

Bose, B. K. (2017). Artificial Intelligence Techniques in Smart Grid and Renewable Energy Systems - Some Example Applications. *Proceedings of the IEEE*, *105*(11), 2262–2273. https://doi.org/10.1109/JPROC.2017.2756596

Boureau, Y. L., Bach, F., LeCun, Y., & Ponce, J. (2010). Learning mid-level features for recognition. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2559–2566. https://doi.org/10.1109/CVPR.2010.5539963

Breiman, L. (2001). Random Forests. *Machine Learning*, (45), 5–32.

Cadík, M., & Vaší˘, J. (2015). Camera Elevation Estimation from a Single Mountain Landscape Photograph.

Calonder, M., Lepetit, V., Strecha, C., & Fua, P. (2010). BRIEF: Binary robust independent elementary features. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *6314 LNCS*(PART 4), 778–792. https://doi.org/10.1007/978-3-642-15561-1_56/COVER

Cao, S., & Snavely, N. (2015). Graph-Based Discriminative Learning for Location Recognition. *International Journal of Computer Vision*, *112*(2), 239–254. https://doi.org/10.1007/s11263-014-0774-9

Chang, M. H., Pyun, J. Y., Ahmad, M. B., Chun, J. H., & Park, J. A. (2005). Modified Color Co-occurrence Matrix for Image Retrieval. *Advances in Natural Computation. Lecture Notes in Computer Science*, *3611*, 43–50.

Chaudhuri, B. B., & Sarkar, N. (1995). Texture Segmentation Using Fractal Dimension. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *17*(1), 72–77. https://doi.org/10.1109/34.368149

Chen, D. M., Baatz, G., & Tsai, S. S. (2011). City-Scale Landmark Identification on Mobile Devices. *CVPR 2011*, 737–744.

Chen, Z., Jacobson, A., Sünderhauf, N., Upcroft, B., Liu, L., Shen, C., & Reid, I. (2017). Deep Learning Features at Scale for Visual Place Recognition. *International Conference on Robotics and Automation (ICRA)*, *1*, 1–8.

Ciocca, G., Cusano, C., Santini, S., & Schettini, R. (2013). High-level features for image indexing and retrieval. In *Encyclopedia of Information Science and Technology*. IGI Global.

Cocco, S., Monasson, R., & Zamponi, F. (2022). Supervised learning: classification with neural networks. *From Statistical Physics to Data-Driven Modelling*, 137–160. https://doi.org/10.1093/OSO/9780198864745.003.0007

Council, N. R. (1991). *Spatial Statistics and Digital Image Analysis*. Washington, DC: Press, The National Academies. https://doi.org/10.17226/1783

Crandall, D. J., Li, Y., Lee, S., & Huttenlocher, D. P. (2016). Recognizing Landmarks in Large-Scale Social Image Collections. *Large-Scale Visual Geo-Localization, Advances in Computer Vision and Pattern Recognition*, 121–144. https://doi.org/10.1007/978-3-319-25781-5

Cross, G. R., & Jain, A. K. (1983). Markov Random Field Texture Models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *PAMI-5*(1), 25–39. https://doi.org/10.1109/TPAMI.1983.4767341

Dewan, J. H., & Thepade, S. D. (2020). Image Retrieval Using Low Level and Local Features Contents : A Comprehensive Review. *Applied Computational Intelligence and Soft Computing*, *2020*, 20. https://doi.org/https://doi.org/10.1155/2020/8851931

Djouadi, M., & Kholladi, M. (2022). Improving Street View Image Classification Using Pre-trained CNN Model Extracted Features. *Periodica Polytechnica Electrical Engineering and Computer Science*, *66*(4), 370–379.

Domingos, P., & Pazzani, M. (1997). On the Optimality of the Simple Bayesian Classifier under Zero-One Loss. *Machine Learning*, *29*(2–3), 103–130. https://doi.org/10.1023/A:1007413511361/METRICS

Donoser, M., & Bischof, H. (2006). Efficient Maximally Stable Extremal Region (MSER) tracking. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, *1*, 553–560. https://doi.org/10.1109/CVPR.2006.107

Erhan, D., Courville, A., Bengio, Y., & Vincent, P. (2010, March 31). Why does unsupervised pre-training help deep learning? *Journal of Machine Learning Research*. JMLR Workshop and Conference Proceedings. Retrieved from https://proceedings.mlr.press/v9/erhan10a.html

Fan, J., Gao, Y., Luo, H., & Xu, G. (2004). Automatic image annotation by using concept-sensitive salient objects for image content representation. *Proceedings of Sheffield SIGIR - Twenty-Seventh Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 361–368. https://doi.org/10.1145/1008992.1009055

Fu, K. S. (1977). *Syntactic Pattern Recognition, Applications*.

Galanis, N. I., Vafiadis, P., Mirzaev, K. G., & Papakostas, G. A. (2022). Convolutional Neural Networks: A Roundup and Benchmark of Their Pooling Layer Variants. *Algorithms 2022, Vol. 15, Page 391*, *15*(11), 391. https://doi.org/10.3390/A15110391

Glistrup, M., Rudinac, S., & Jónsson, B. Þ. (2022). Urban Image Geo-Localization Using Open Data on Public Spaces. *Proceedings of 19th International Conference on Content-Based Multimedia Indexing*, 50–56. https://doi.org/10.1145/3549555.3549589

Gonzalez, R. C., & Woods, R. E. (2018). *Digital Image Processing* (4th editio). New York, NY: Pearson Education.

Gordo, A. (2015). Supervised Mid-Level Features for Word Image Representation. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Gron, P., Obozinski, G., & Sivic, J. (2013). Learning and calibrating per-location classifiers for visual place recognition. https://doi.org/10.1109/CVPR.2013.122

Guo, G., Wang, H., Bell, D., Bi, Y., & Greer, K. (2003). KNN model-based approach in classification. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *2888*, 986–996. https://doi.org/10.1007/978-3-540-39964-3_62/COVER

Haenlein, M., & Kaplan, A. (2019). A Brief History of Artificial Intelligence:, 1–10. https://doi.org/10.1177/0008125619864925

Harris, C., & Stephens, M. (1988). A combined corner and edge detector. *Proceedings Of the 4th Alvey Vision Conference*, 147–152.

Hays, J., & Efros, A. A. (2008). IM2GPS: Estimating geographic information from a single image. *26th IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, *05*. https://doi.org/10.1109/CVPR.2008.4587784

Hays, J., & Efros, A. A. (2015). Large-Scale Image Geolocalization. *In: Choi J., Friedland G. (Eds) Multimodal Location Estimation of Videos and Images. Springer, Cham*, 1–191. https://doi.org/10.1007/978-3-319-09861-6

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, *2016-December*, 770–778. https://doi.org/10.1109/CVPR.2016.90

Hervé, N., & Boujemaa, N. (2007). Image annotation: Which approach for realistic databases? *Proceedings of the 6th ACM International Conference on Image and Video Retrieval, CIVR 2007*, 170–177. https://doi.org/10.1145/1282280.1282310

Hiba, C., Hamid, Z., Jadida, E., & ALHEYANE Omar, M. (2016). Bag of Features Model Using the New Approaches: A Comprehensive Study. *International Journal of Advanced Computer Science and Applications*, *7*(1). https://doi.org/10.14569/IJACSA.2016.070132

Hu, M. K. (1962). Visual Pattern Recognition by Moment Invariants. *IRE Transactions on Information Theory*, *8*(2), 179–187. https://doi.org/10.1109/TIT.1962.1057692

Huang, B., Huan, Y., Xu, L. Da, Zheng, L., & Zou, Z. (2019). Automated trading systems statistical and machine learning methods and hardware implementation : a survey, *7575*. https://doi.org/10.1080/17517575.2018.1493145

Huang, C., & Wang, G. (2006). Method of image retrieval based on color coherence vector. *Computer Engineering*, *32*(2), 194–199.

Huang, J., Kumar, S. R., Mitra, M., Zhu, W.-J., & Zabih, R. (1997). Image Indexing Using Color Correlograms. *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 762–768.

Irschara, A., Zach, C., Frahm, J. M., & Bischof, H. (2009). From structure-from-motion point clouds to fast location recognition. *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, CVPR Workshops 2009*, *2009 IEEE*, 2599–2606. https://doi.org/10.1109/CVPRW.2009.5206587

Isabelle Guyon, Steve Gunn, Masoud Nikravesh, L. A. Z. (2006). *Feature Extraction: Foundations and Applications*. Springer.

Islam, M. M., Zhang, D., & Lu, G. (2008). A geometric method to compute directionality features for texture images. *Proceedings of the International Conference on Multimedia and Expo*, (3), 1521–1524.

Islam, S., Elmekki, H., Elsebai, A., Bentahar, J., Drawel, N., Rjoub, G., & Pedrycz, W. (2024). A comprehensive survey on applications of transformers for deep learning tasks. *Expert Systems With Applications*, *241*(June 2023), 1–48. https://doi.org/10.1016/j.eswa.2023.122666

James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). *An Introduction to Statistical Learning*. New York, NY: Springer.

Johns, J. M., Rounds, J., & Henry, M. J. (2017). Multi-modal Geolocation Estimation Using Deep Neural Networks, 1–12. Retrieved from http://arxiv.org/abs/1712.09458

Kendall, A., & Cipolla, R. (2017). Geometric loss functions for camera pose regression with deep learning.

Kendall, A., & College, K. (2016). PoseNet: A Convolutional Network for Real-Time 6-DOF Camera Relocalization.

Kim, H. J., Dunn, E., & Frahm, J. M. (2015). Predicting good features for image geo-localization using per-bundle VLAD. *Proceedings of the IEEE International Conference on Computer Vision*, *2015 Inter*, 1170–1178. https://doi.org/10.1109/ICCV.2015.139

Knopp, J., Sivic, J., & Pajdla, T. (2010). Avoiding confusing features in place recognition. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *6311 LNCS*(PART 1), 748–762. https://doi.org/10.1007/978-3-642-15549-9_54

Kovalev, V., & Volmer, S. (1998). Color Co-Occurrence Descriptors for Querying – by – Example. *Proceedings 1998 MultiMedia Modeling. MMM'98 (Cat. No.98EX200)*, 1–7.

Krichen, M. (2023). Convolutional Neural Networks : A Survey, 1–41.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). ImageNet Classification with Deep Convolutional Neural Networks, 1–9. https://doi.org/10.1201/9781420010749

Latif, A., Rasheed, A., Sajid, U., Ahmed, J., Ali, N., Ratyal, N. I., … Khalil, T. (2019). Content-based image retrieval and feature extraction: A comprehensive review. *Mathematical Problems in Engineering*, *2019*, 1–22. https://doi.org/10.1155/2019/9658350

Lazebnik, S., Schmid, C., & Ponce, J. (2006). Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, *2*, 2169–2178. https://doi.org/10.1109/CVPR.2006.68

Lazebnik, S., Schmid, C., & Ponce, J. (2009). Spatial pyramid matching. *Object Categorization: Computer and Human Vision Perspectives*, *9780521887*, 401–415. https://doi.org/10.1017/CBO9780511635465.022

Lee, K. L., & Chen, L. H. (2005). An efficient computation method for the texture browsing descriptor of MPEG-7. *Image and Vision Computing*, *23*(5), 479–489. https://doi.org/10.1016/J.IMAVIS.2004.12.002

Leszek Cieplinski, Kim, M., Ohm, J.-R., Pickering, M., & Yamada, A. (2001). *ISO/IEC 15938-3/FCD Information Technology – Multimedia Content Description Interface – Part 3 Visual*. Singapore.

Li, Y., Crandall, D. J., & Huttenlocher, D. P. (2009). Landmark Classification in Large-scale Image Collections, (Iccv), 2–9.

Li, Y., Snavely, N., Huttenlocher, D. P., & Fua, P. (2016). Worldwide Pose Estimation Using 3D Point Clouds, 147–163. https://doi.org/10.1007/978-3-319-25781-5_8

Lin, T. Y., Belongie, S., & Hays, J. (2013). Cross-view image geolocalization. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 891–898. https://doi.org/10.1109/CVPR.2013.120

Liu, F., & Picard, R. W. (1996). Periodicity, directionality, and randomness: Wold features for image modeling and retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *18*(7), 722–733. https://doi.org/10.1109/34.506794

Long, B., Konkle, T., Cohen, M. A., & Alvarez, G. A. (2016). Mid-Level Perceptual Features Distinguish Objects of Different Real-World Sizes. *Journal of Experimental Psychology: General*, *145*(1), 95–109.

Long, F., Zhang, H., & Feng, D. D. (2003). Fundamentals of Content-Based Image Retrieval. *(Eds) Multimedia Information Retrieval and Management. Signals and Communication Technology. Springer*, 1–26.

Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, *60*(2), 91–110. https://doi.org/10.1023/B:VISI.0000029664.99615.94/METRICS

Lu, G., Yan, Y., Ren, L., Song, J., Sebe, N., & Kambhamettu, C. (2015). Localize me anywhere, anytime: A multi-task point-retrieval approach. *Proceedings of the IEEE International Conference on Computer Vision*, *2015 Inter*, 2434–2442. https://doi.org/10.1109/ICCV.2015.280

Lu, Z.-M., Li, S.-Z., & Burkhardt, H. (2006). A content-based image retrieval scheme in JPEG compressed domain. *International Journal of Innovative Computing, Infor-Mation and Control*, *2*(4), 831–839.

Luo, J., & Savakis, A. (2001). Indoor vs outdoor classification of consumner photographs using low-level and semantic features. *IEEE International Conference on Image Processing*, *2*, 745–748. https://doi.org/10.1109/ICIP.2001.958601

Manjunath, B. S., & Ma, W. Y. (1996). Texture features for browsing and retrieval of image data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *18*(8), 837–842. https://doi.org/10.1109/34.531803

Manjunath, B. S., Ohm, J. R., Vasudevan, V. V., & Yamada, A. (2001). Color and texture descriptors. *IEEE Transactions on Circuits and Systems for Video Technology*, *11*(6), 703–715. https://doi.org/10.1109/76.927424

Masone, C., & Caputo, B. (2021). A Survey on Deep Visual Place Recognition, *9*.

Matas, J., Chum, O., Urban, M., & Pajdla, T. (2004). Robust wide-baseline stereo from maximally stable extremal regions. *Image and Vision Computing*, *22*(10), 761–767. https://doi.org/10.1016/J.IMAVIS.2004.02.006

Materka, A., & Strzelecki, M. (1998). Texture Analysis Methods – A Review. *Technical University of Lodz, Institute of Electronics*, 1–33.

Muller-Budack, E., Pustu-Iren, K., & Ewerth, R. (2018). Geolocation Estimation of Photos Using a Hierarchical Model and Scene Classification Eric. *Springer Nature Switzerland AG 2018*, 575–592. https://doi.org/10.1007/978-3-030-01258-8 35

Murphy, K. P. (2006). Naive Bayes classifiers, 1–8.

Najafabadi, M. M., Villanustre, F., Khoshgoftaar, T. M., Seliya, N., Wald, R., & Muharemagic, E. (2015). Deep learning applications and challenges in big data analytics. *Journal of Big Data*, *2*(1), 1–21. https://doi.org/10.1186/S40537-014-0007-7/METRICS

Neyshabur, B., Bhojanapalli, S., McAllester, D., & Srebro, N. (2017). Exploring Generalization in Deep Learning. *Advances in Neural Information Processing Systems*, *2017-December*, 5948–5957. Retrieved from https://arxiv.org/abs/1706.08947v2

Niu, D., Zhao, X., Lin, X., & Zhang, C. (2020). A novel image retrieval method based on multi-features fusion. *Signal Processing: Image Communication*, *87*, 115911. https://doi.org/10.1016/J.IMAGE.2020.115911

Noh, H., Sim, J., & Weyand, T. (2018). Large-Scale Image Retrieval with Attentive Deep Local Features.

O'Hara, S., & Draper, B. A. (2011). Introduction to the Bag of Features Paradigm for Image Classification and Retrieval, (January 2011), 1–25.

Park, S. B., Lee, J. W., & Kim, S. K. (2004). Content-based image classification using a neural network. *Pattern Recognition Letters*, *25*(September 2002), 287–300. https://doi.org/10.1016/j.patrec.2003.10.015

Pass, G., & Zabih, R. (1996). Histogram refinement for content-based image retrieval. *IEEE Workshop on Applications of Computer Vision - Proceedings*, 96–102. https://doi.org/10.1109/acv.1996.572008

Paul V C Hough. (1960). Method and means for recognizing complex patterns.

Peddada, A. V, & Hong, J. (2015). Geo-Location Estimation with Convolutional Neural Networks, 1–9. Retrieved from http://cs231n.stanford.edu/reports/CS231N_Final_Report_amanivp_jamesh93.pdf

Prijono, B. (n.d.). Student Notes: Convolutional Neural Networks (CNN) Introduction.

Rish, I. (2001). An empirical study of the naive Bayes classifier. *IJCAI Workshop on Empirical Methods in Artificial Intelligence*, 41–46.

Rosten, E., & Drummond, T. (2006). Machine learning for high-speed corner detection. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *3951 LNCS*, 430–443. https://doi.org/10.1007/11744023_34/COVER

Rublee, E., Rabaud, V., Konolige, K., & Bradski, G. (2011). ORB: An efficient alternative to SIFT or SURF. *Proceedings of the IEEE International Conference on Computer Vision*, 2564–2571. https://doi.org/10.1109/ICCV.2011.6126544

Rui, Y., & Huang, T. S. (1999). Image Retrieval : Current Techniques , Promising Directions , and Open Issues. *Journal of Visual Communication and Image Representation*, *62*(10), 39–62.

Salembier, P., & Manjunath, B. S. (2002). *Introduction to MPEG 7: Multimedia Content Description Language* (1st editio). Wiley.

Salim Malek. (2018). Deep Neural Network Models For Image Classification and Regression. *University of Trento*, 1–98.

Sattler, T., Group, C. G., Group, C. V., Weyand, T., Leibe, B., & Kobbelt, L. (2012). Image Retrieval for Image-Based Localization Revisited.

Sattler, T., Havlena, M., Schindler, K., & Pollefeys, M. (2016). Large-scale location recognition and the geometric burstiness problem. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, *2016-Decem*, 1582–1590. https://doi.org/10.1109/CVPR.2016.175

Sattler, T., Leibe, B., & Kobbelt, L. (2011). Fast Image-Based Localization using Direct 2D-to-3D Matching.

Saurer, O., Baatz, G., Köser, K., Ladický, L., & Pollefeys, M. (2016). Image Based Geo-localization in the Alps. *International Journal of Computer Vision*, *116*(3), 213–225. https://doi.org/10.1007/s11263-015-0830-0

Schindler, G., Brown, M., & Szeliski, R. (2007). City-scale location recognition. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. https://doi.org/10.1109/CVPR.2007.383150

Selvapattu, S., & Nair, S. R. (2022). High and Low-Level Classi cation based Features for Content-based Image Retrieval in Privacy- Preserving. *Journal of Cloud Computing-Advances Systems and Applications*, 1–29.

Seo, P. H., Weyand, T., Sim, J., & Han, B. (2018). Enhancing Image Geolocalization by Combinatorial Partitioning of Maps.

Shao, H., Wu, Y., Cui, W., & Zhang, J. (2008). Image retrieval based on MPEG-7 dominant color descriptor. *Proceedings of the 9th International Conference for Young Computer Scientists, ICYCS 2008*, 753–757. https://doi.org/10.1109/ICYCS.2008.89

Sharma, R., & Singh, E. N. (2014). Comparative Study of Different Low Level Feature Extraction Techniques, *3*(4), 1454–1460.

Sharma, S., & Guleria, K. (2022). Deep Learning Models for Image Classification : Comparison and Applications. *2022 2nd International Conference on Advance Computing and Innovative Technologies in Engineering (ICACITE)*, 1733–1738. https://doi.org/10.1109/ICACITE53722.2022.9823516

Shi, J., & Tomasi, C. (1994). Good features to track. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 593–600. https://doi.org/10.1109/CVPR.1994.323794

Shim, S.-O., & Choi, T. (2003). Image indexing by modified color co-occurrence matrix. *Proceedings 2003 International Conference on Image Processing (Cat. No.03CH37429)*, 493–496.

Simonyan, K., & Zisserman, A. (2014). Very Deep Convolutional Networks for Large-Scale Image Recognition. *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*. Retrieved from https://arxiv.org/abs/1409.1556v6

Smith, J. R., & Chang, S. (1996). Tools and Techniques for Color Image Retrieval. *SPIE Storage and Retrieval for Image and Video Databases IV*, *2670*, 426–437.

Sonka, M., Hlavac, V., & Boyle, R. (2015). *Image processing, analysis and machine vision* (4th Editio). Global Engineering: Timothy L. Anderson.

Starck, J. L., Candès, E. J., & Donoho, D. L. (2002). The curvelet transform for image denoising. *IEEE Transactions on Image Processing*, *11*(6), 670–684. https://doi.org/10.1109/TIP.2002.1014998

Stricker, M., & Orengo, M. (1995). Similarity of Color Images. *Proceedings Spie Storage & Retrieval for Image & Video Databases*, *2420*, 381–392.

Sumana, I. J., Islam, M. M., Zhang, D., & Lu, G. (2008). Content based image retrieval using curvelet transform. *Proceedings of the 2008 IEEE 10th Workshop on Multimedia Signal Processing, MMSP 2008*, 11–16. https://doi.org/10.1109/MMSP.2008.4665041

Suresh, S., Chodosh, N., & Abello, M. (2018). DeepGeo: Photo localization with deep neural network. *ArXiv*, 2–8.

Swain, M. J., & Ballard, D. H. (1991). Color Indexing. *International Journal of Computer Vision*, *7*(1), 11–32.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., … Rabinovich, A. (2015). Going deeper with convolutions. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, *07-12-June-2015*, 1–9. https://doi.org/10.1109/CVPR.2015.7298594

Szeliski, R. (2022). Computer Vision: Algorithms and Applications, 2nd Edition. *Springer*, Cap. 1. Retrieved from https://www.wiley.com/en-in/Machine+Vision+Algorithms+and+Applications%2C+2nd+Edition-p-9783527413652

Taherdoost, H. (2022). Classification of Machine Learning Algorithms. *Advances in Data Computing, Communication and Security, Lecture Notes on Data Engineering and Communications Technologies 106*, 417–422.

Tamura, H., Mori, S., & Yamawaki, T. (1978). Textural Features Corresponding Visual Perception. *IEEE TRANSACTIONS ON SYSTEMS. MAN, AND CYBERNETICS*, *75*(6), 460–473.

Tannaz, S., & Sedghi, T. (2018). Image Retrieval Using Dynamic Weighting of Compressed High Level Features Framework with LER Matrix. *Iranian Journal of Electrical & Electronic Engineering*, *14*(2), 153–161.

The vprice challenge 2015 visual place recognition in changing environments. (n.d.). Retrieved September 14, 2018, from https://roboticvision.atlassian.net/wiki/spaces/PUB/pages/14188617/The+VPRiCE+Challenge+2015+Visual+Place+Recognition+in+Changing+Environments

Thomee, B., Shamma, D. A., Friedland, G., Poland, D., & Borth, D. (n.d.). YFCC100M : The New Data in Multimedia Research.

Tian, D. P. (2013). A Review on Image Feature Extraction and Representation Techniques. *International Journal of Multimedia and Ubiquitous Engineering*, *8*(4), 385–396.

Tian, Y., Chen, C., & Shah, M. (2017). Cross-View Image Matching for Geo-localization in Urban Environments, 4321–4329.

Torii, A., & Pajdla, T. (2013). Visual Place Recognition with Repetitive Structures, 883–890. https://doi.org/10.1109/CVPR.2013.119

Tsung-Yi Lin, Yin Cui, Serge Belongie, J. H. (n.d.). Learning Deep Representations for Ground-to-Aerial Geolocalization. https://doi.org/10.1007/BF00522958

Tsung-Yi Lin, Yin Cui, Serge Belongie, J. H. (2015). Learning Deep Representations for Ground-to-Aerial Geolocalization, 5007–5015.

TUCERYAN, M., & JAIN, A. K. (1993). TEXTURE ANALYSIS. *Handbook of Pattern Recognition and Computer Vision*, 235–276. https://doi.org/10.1142/9789814343138_0010

Tuytelaars, T., & Mikolajczyk, K. (2007). Local invariant feature detectors: A survey. *Foundations and Trends in Computer Graphics and Vision*, *3*(3), 177–280. https://doi.org/10.1561/0600000017

Tzeng, E., Zhai, A., Clements, M., Townshend, R., & Zakhor, A. (2013). User-driven geolocation of untagged desert imagery using digital elevation models. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, 237–244. https://doi.org/10.1109/CVPRW.2013.42

Vailaya, A., Figueiredo, M. A. T., Jain, A. K., & Zhang, H. J. (2001). Image classification for content-based indexing. *IEEE Transactions on Image Processing*, *10*(1), 117–130. https://doi.org/10.1109/83.892448

Verma, P., Charan, C., Fernando, X., & Ganesan, S. (2022). Advances in Data Computing , Communication and Security. In S. Fatos Xhafa, Technical University of Catalonia, Barcelona (Ed.), *International Conference on Advances in Data Computing, Communication and Security (I3CS2021)* (p. 438). Springer Nature Singapore.

Vo, N., Jacobs, N., & Hays, J. (2017). Revisiting IM2GPS in the Deep Learning Era. *Proceedings of the IEEE International Conference on Computer Vision*, *2017-Octob*(1), 2640–2649. https://doi.org/10.1109/ICCV.2017.286

Walch, F., Leal-taix, C. H. L., Sattler, T., & Cremers, S. H. D. (2017). Image-based localization using LSTMs for structured feature correlation.

Wang, J. Z., Li, J., & Wiederhold, G. (2001). SIMPLIcity: Semantics-sensitive integrated Matching for Picture Libraries. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *23*(9), 947–963. https://doi.org/10.1109/34.955109

Wang, P. (2019). On Defining Artificial Intelligence, *10*(2), 1–37. https://doi.org/10.2478/jagi-2019-0002

Wang, Q., Zhou, C., & Xu, N. (2017). Street view image classification based on convolutional neural network. *Proceedings of 2017 IEEE 2nd Advanced Information Technology, Electronic and Automation Control Conference, IAEAC 2017*, 1439–1443. https://doi.org/10.1109/IAEAC.2017.8054251

Wang, S., Han, K., & Jin, J. (2019). Review of image low-level feature extraction methods for content-based image retrieval. *Sensor Review*, *39*(6), 783–809. https://doi.org/10.1108/SR-04-2019-0092

Wang, X., Zhang, B., & Yang, H. (2014). Content-based image retrieval by integrating color and texture features. *Multimed Tools Appl*, (04), 545–569. https://doi.org/10.1007/s11042-012-1055-7

Weyand, T., Kostrikov, I., & Philbin, J. (2016). Planet - photo geolocation with convolutional neural networks. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *9912 LNCS*, 37–55. https://doi.org/10.1007/978-3-319-46484-8_3

Weyand, T., & Leibe, B. (2015). Visual landmark recognition from Internet photo collections : A large-scale evaluation q. *COMPUTER VISION AND IMAGE UNDERSTANDING*. https://doi.org/10.1016/j.cviu.2015.02.002

What Is Deep Learning? (n.d.). Retrieved December 15, 2023, from https://www.mathworks.com/discovery/deep-learning.html

Workman, S., Souvenir, R., & Jacobs, N. (2015). Wide-area image geolocalization with aerial reference imagery. *Proceedings of the IEEE International Conference on Computer Vision*, *2015 Inter*(2), 3961–3969. https://doi.org/10.1109/ICCV.2015.451

Yan, F. (2016). Deep Learning.

Yang, C., Yang, Z., Hou, J., & Su, Y. (2021). A Lightweight Full Homomorphic Encryption Scheme on Fully-connected Layer for CNN Hardware Accelerator achieving Security Inference. *2021 28th IEEE International Conference on Electronics, Circuits, and Systems, ICECS 2021 - Proceedings*. https://doi.org/10.1109/ICECS53924.2021.9665520

Yavlinsky, A., Schofield, E., & Stefan, R. (2005). Features and Robust Nonparametric Density Estimation. *Proceedings of the International Conference on Image and Video Retrieval, Singapore*, 507–517.

Zamir, Amir R., Hakeem, A., Van Gool, L., Shah, M., & Szeliski, R. (2016). *Introduction to Large-Scale Visual Geo-localization*. https://doi.org/10.1007/978-3-319-25781-5_1

Zamir, Amir Roshan. (2014). GPS-Tag Refinement using Random Walks with an Adaptive Damping Factor, 4280–4287. https://doi.org/10.1109/CVPR.2014.545

Zamir, Amir Roshan, & Shah, M. (2010). Accurate image localization based on google maps street view. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 6314 LNCS*(PART 4), 255–268. https://doi.org/10.1007/978-3-642-15561-1_19

Zamir, Amir Roshan, & Shah, M. (2014). Image geo-localization based on multiplenearest neighbor feature matching usinggeneralized graphs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *36*(8), 1546–1558. https://doi.org/10.1109/TPAMI.2014.2299799

Zeiler, M. D., & Fergus, R. (2014). Visualizing and Understanding Convolutional Networks. *Computer Vision–ECCV 2014*, *8689*(PART 1), 818–833. https://doi.org/10.1007/978-3-319-10590-1_53

Zemene, E., Tesfaye, Y. T., Idrees, H., Prati, A., Pelillo, M., & Shah, M. (2019). Large-Scale Image Geo-Localization Using Dominant Sets. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *41*(1), 148–161. https://doi.org/10.1109/TPAMI.2017.2787132

Zhang, Caiming, & Lu, Y. (2021). Study on artificial intelligence : The state of the art and future prospects. *Journal of Industrial Information Integration*, *23*(March), 100224. https://doi.org/10.1016/j.jii.2021.100224

Zhang, Chi, & Huang, L. (2014). Content-Based Image Retrieval Using Multiple Features. *Journal of Computing and Information Technology - CIT 22*, (Special Issue), 1–10.

Zhang, D., Islam, M. M., & Lu, G. (2012). A review on automatic image annotation techniques. *Pattern Recognition*, *45*(1), 346–362. https://doi.org/10.1016/j.patcog.2011.05.013

Zhang, D., Wong, A., Indrawan-Santiago, M., & Lu, G. (2000). Content-based image retrieval using Gabor texture features. *Proceedings of the First IEEE Pacific- Rim Conference on Multimedia*, 392–395.

Zhang, R., Zhang, Z., Li, M., Ma, W. Y., & Zhang, H. J. (2005). A probabilistic semantic model for image annotation and multi-modal image retrieval. *Proceedings of the IEEE International Conference on Computer Vision*, *1*, 846–851. https://doi.org/10.1109/ICCV.2005.16

Zheng, Y.-T., Ming Zhao, Yang Song, Adam, H., Buddemeier, U., Bissacco, A., … Neven, H. (2010). Tour the world: Building a web-scale landmark recognition engine, 1085–1092. https://doi.org/10.1109/cvpr.2009.5206749