Document Image Layout Segmentation and Applications

Thesis submitted in partial fulfillment
of the requirements for the degree of

Doctor of Philosophy
in
Electronics and Communication Engineering

by

Jobin K.V.
201350919
jobin.kv@research.iiit.ac.in

International Institute of Information Technology
Hyderabad - 500 032, INDIA
April, 2025

International Institute of Information Technology
Hyderabad, India

CERTIFICATE

It is certified that the work contained in this thesis, titled "Document Image Layout Segmentation and Applications" by Jobin K.V, has been carried out under my supervision and is not submitted elsewhere for a degree.

Date

Adviser: Prof. C.V. Jawahar

To

My Parents

Acknowledgments

I express my heartfelt gratitude to Professor C. V. Jawahar, my advisor, for his invaluable guidance and the numerous opportunities he has provided me, including mentorship, participation in workshops and conferences, and engaging in research collaborations. I appreciate his belief in my capabilities and unwavering support during challenging moments. I thank the Visvesvarayya PhD Fellowship program by the Government of India for generously supporting my research financially. I thank Dr. Jiji C.V. of the College of Engineering, Trivandrum, for inspiring me to pursue a PhD. I also thank Dr. Ajoy Mondal and Dr. Anand Mishra for their mentorship and collaboration in my research.

I would like to convey my appreciation to the research community, encompassing my fellow lab-mates, researchers from other labs within IIIT, members of my thesis committee, and the reviewers of my journal and conference proceedings. Their earnest contributions in engaging, evaluating, and offering constructive feedback on my work are sincerely acknowledged.

During the course of my PhD, I have come to know many incredible people who have contributed greatly to my professional and personal growth. In addition to my collaborators, I also thank Yashaswi, Aditya, Sourabh D, Aniket, Pritish, Vijay, Avijit, Seshadri, Sachin and other lab-mates who I met during the early years of my PhD, for their technical help and for creating a vibrant and joyful place of research.

I extend my heartfelt appreciation to the administrative team at CVIT, comprising Mr. Satyanarayana, the late Mr. Phaneendra, Siva, Silar, Ram, Sireesha, Rohita, Aradhana, Varun, Sony, Ann, Cecilia, Mahender, and Mahathi. I am also grateful to the administrative staff at IIIT, namely Ms. Prathima, Ms. Pushpalatha, Mr. Y Kishore, Mr. Appaji, Mr. Murthy, Mr. Satheesh, and Mr. Mahender. Their dedicated support has been instrumental in ensuring the seamless progression of the research student's needs. Thank you for your valuable contributions to this process.

A sincere appreciation to the Malayali community at IIIT, including Praveen, Devadath, Varun, Minesh, Aquib, Mithun, Shajil, Jerin, Vinitha, Litton, Ratheesh, Siji, Thrupthi, and others. I will always cherish the enduring connections with the exceptional faculty and friends from my previous alma maters, the College of Engineering Trivandrum and Adoor.

I want to express my sincere gratitude to my parents, brother, and parents-in-law for their steadfast support and enduring confidence in me. I am especially thankful to my wife, Kitty, and my son, Naiden, whose continuous encouragement has been a constant source of motivation and strength. My thesis also serves as a modest tribute to the teachers who guided and instructed me throughout this journey.

Abstract

A document has information emerging out of the existence of various physical entities or regions such as headings, paragraphs, figures, captions, tables, and backgrounds along with the textual content. To decipher the information in a document, a human reader uses a variety of additional cues, such as context, conventions, and information about language, script, location, and a complex reasoning process. Millions of documents are created and distributed daily over the Internet and printed media. Understanding, analyzing, sorting, and comparing a massive collection of documents in a limited time is a hectic job for humans. Here, the automatic document image understanding systems (DIUS) help humans do this tedious task within a limited time. The DIUS have typically a document image layout segmentation module and information extraction modules. This thesis focuses on the challenging problems related to document image layout segmentation in various types of documents and their applications.

In this thesis, first, we analyse the various document images using deep features. The deep features are the features extracted using a pretrained deep neural network. To study deep texture features, we propose a deep network architecture that independently learns texture patterns, discriminative patches, and shapes to solve various document image analysis tasks. The considered tasks are document image classification, genre identification from book covers, scientific document figure classification, and script identification. The presented network learns global, texture, and discriminative features and combines them judicially based on the nature of the problems. We compare the performance of the proposed approach with state-of-the-art techniques on multiple publicly available datasets such as Book covers, RVL-CDIP, CVSI and DocFigure. Experiments show that our approach obtains significantly better performance over state-of-the-art for all tasks.

Next, we focus on the problem of document image layout segmentation and propose a solution for a class of document images, including historical, scientific, and classroom slide document images. The historical document image segmentation problem is modeled as a pixel labeling task where each pixel in the document image is classified into one of the predefined labels, such as text, comment, decoration, and background. The method first extracts deep features from the superpixels of the document image. Then, we learn SVM classifier using these features and segment the document image. The pyramid pooling module is used to extract the logical regions of scientific document images. In the classroom slide images, the logical regions are distributed based on the location of the image. To utilize the location of the logical regions for slide image segmentation, we propose the architecture, Classroom Slide Segmentation Network (CSSN). The unique attributes of this architecture differ from most other

semantic segmentation networks. We validate the performance of our segmentation architecture using publicly available benchmark datasets.

Next, we analyze the output regions of document layout segmentation. Figures used in the documents are the complex regions to decipher the information by a DIUS. Hence, document figure classification (DFC) is an important stage of the DIUS. The design of a DFC system required well-defined figure categories and datasets. Existing datasets related to the classification of figures in the document images are limited in size and category. We introduce a scientific figure classification dataset named `DocFigure`. The dataset consists of 33K annotated figures of 28 different categories present in the document images, which correspond to scientific articles published in last several years. Manual annotation of such a large number (33K) of figures is time-consuming and cost-ineffective. We design a web-based annotation tool that can efficiently assign category labels to many figures with the minimum effort of human annotators. To benchmark our generated dataset on the classification task, we propose three baseline classification techniques using the deep feature, deep texture feature, and both. Our analysis found that the combination of both deep and texture features is more effective for document figure classification tasks than individual features.

Finally, we propose the application backed by the research of this thesis. Slide presentations are an effective and efficient tool for classroom communication used by the teaching community. However, this teaching model can be challenging for blind and visually impaired (VI) students as such students require personal human assistance to understand the presented slide. This shortcoming motivates us to design a Classroom Slide Narration System (CSNS) that generates audio descriptions corresponding to the slide content. This problem poses an image-to-markup language generation task. Extract logical regions such as title, text, equation, figure, and table from the slide image using CSSN. We extract the content (information) from the slide using four well-established modules: optical character recognition (OCR), figure classification, equation description, and table structure recognizer. With this information, we build a Classroom Slide Narration System (CSNS) to help VI students understand the slide content. The users have given better feedback on the quality output of the proposed CSNS in comparison to existing systems like Facebook's Automatic Alt-Text (AAT) and Tesseract.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

A document is a written or printed record that conveys information, facts, ideas, or instructions. It serves as a means of communication and can take various forms, including text, images, diagrams, charts, and more. Documents have many purposes in personal and professional contexts, such as recording information, sharing knowledge, providing evidence, and facilitating communication. They can be physical, like printed papers, or digital, stored in electronic formats like PDFs, Word documents, spreadsheets, or web pages. Documents are crucial in organizing, preserving, and disseminating information in society. Despite the widespread adoption of electronic communication methods, paper documents such as data entry forms, postal envelopes, and checks remain significant in our daily routines. These physical documents remain highly relevant due to their cost-effectiveness, reliability, secure long-term storage capabilities, widespread availability, and flexibility for data entry. Consequently, the production of paper documents has surged in recent times. Furthermore, government entities and private organizations rely heavily on paper-based documents for information collection. However, manually collecting handwritten or typed information from paper forms and entering it into computer systems is labor-intensive and time-consuming.

Human analysis of a large number of documents comes with several limitations and challenges. Firstly, it is an immensely time-consuming endeavor, mainly when dealing with a substantial volume of documents. Reading, comprehending, and extracting information from each document can be a labor-intensive process that may stretch over extended periods. Additionally, manual analysis is inherently inefficient as humans have limitations in processing vast amounts of data effectively. Fatigue and diminishing attention span can lead to declining accuracy and productivity, potentially resulting in missed details or analysis errors. Moreover, human analysts may introduce subjectivity and bias into their assessments, leading to inconsistent results as different individuals interpret the same information differently. This subjectivity can compromise the quality and reliability of the analysis. Furthermore, scalability becomes a significant issue as the volume of documents increases, necessitating substantial workforce and resource allocation for effective processing. Manual analysis can also become repetitive, increasing the risk of errors or omissions as analysts become fatigued. Data loss is another concern, as human errors or oversights during manual analysis can inadvertently exclude vital information. Manual analysis may not be practical for specific document types or data formats, making it less effective than

automated tools for handling and extracting insights from unstructured text data. The costs associated with employing a large team of analysts for manual analysis is huge, and the quality of analysis can vary depending on the skills and experience of the analysts. Finally, compliance and legal concerns come into play, as organizations may need to adhere to strict requirements when handling documents, and manual analysis may not provide the necessary audit trails and data protection measures. These limitations emphasize the importance of automation in document analysis processes to improve efficiency, accuracy, and scalability while mitigating the challenges associated with manual analysis.

These limitations trigger researchers to invent automated solutions to assist in analyzing and extracting insights from large volumes of documents more efficiently and consistently. In 1982, a groundbreaking document image analysis system found its first practical application within the United States Postal Service (USPS) [1]. This pioneering technology featured the world's first computer-driven single-line optical character reader, capable of swiftly deciphering destination addresses on mail pieces. This monumental achievement marked a significant turning point in mail processing, drastically reducing the need for manual labor in sorting postcards and other mail items. The result of the research on the problem of identifying the handwritten character [2] of given images supported the successful implementation of the system.

Automatic document image analysis has played a pivotal role in streamlining various document-related processes across multiple sectors. Document image analysis is a vibrant field with numerous challenging and active research areas. Among these, optical character recognition (OCR) remains a pivotal focus for printed and handwritten text, diverse languages, math equations, and even music notes, pushing the boundaries of character recognition technology [3–6]. Document layout analysis (DLA) [7, 8] is crucial to developing efficient methods for organizing and parsing documents into meaningful components, enabling more precise analysis and interpretation of their structure and content.

The approaches are categorized into two: the pre-deep learning approach and the deep learning approach. Most pre-deep learning approaches [12–16] are designed based on the binary document image. Here, the image binarization plays a vital role. Hence, various approaches are proposed for image binarization [17–20]. These approaches also fail if the document image has any skew. Most pre-deep learning approaches require a preprocessing stage to remove skew and get a binary image. In addition, most approaches use hyperparameters that depend on the dpi of the document image or the font size. A detailed explanation of these approaches is included in Chapter 2. To overcome this limitation, researchers propose deep learning-based solutions that overcome the dependency of document image binarization, skew corrections and a set of hyperparameters. The deep learning approach for DLA can be categorized into object detection and instance segmentation-based approaches.

In the object detection-based approach, the researchers use object detectors from computer vision literature, such as Faster R-CNN models [21] and Single-Shot MultiBox Detector (SSD) [22] to solve the DLA problem. Liao *et al.,* [23] propose TextBoxes for scene text detection based on Faster R-CNN models [21]. Minouei *et al.,* [24] developed their proposal network to identify the logical regions of

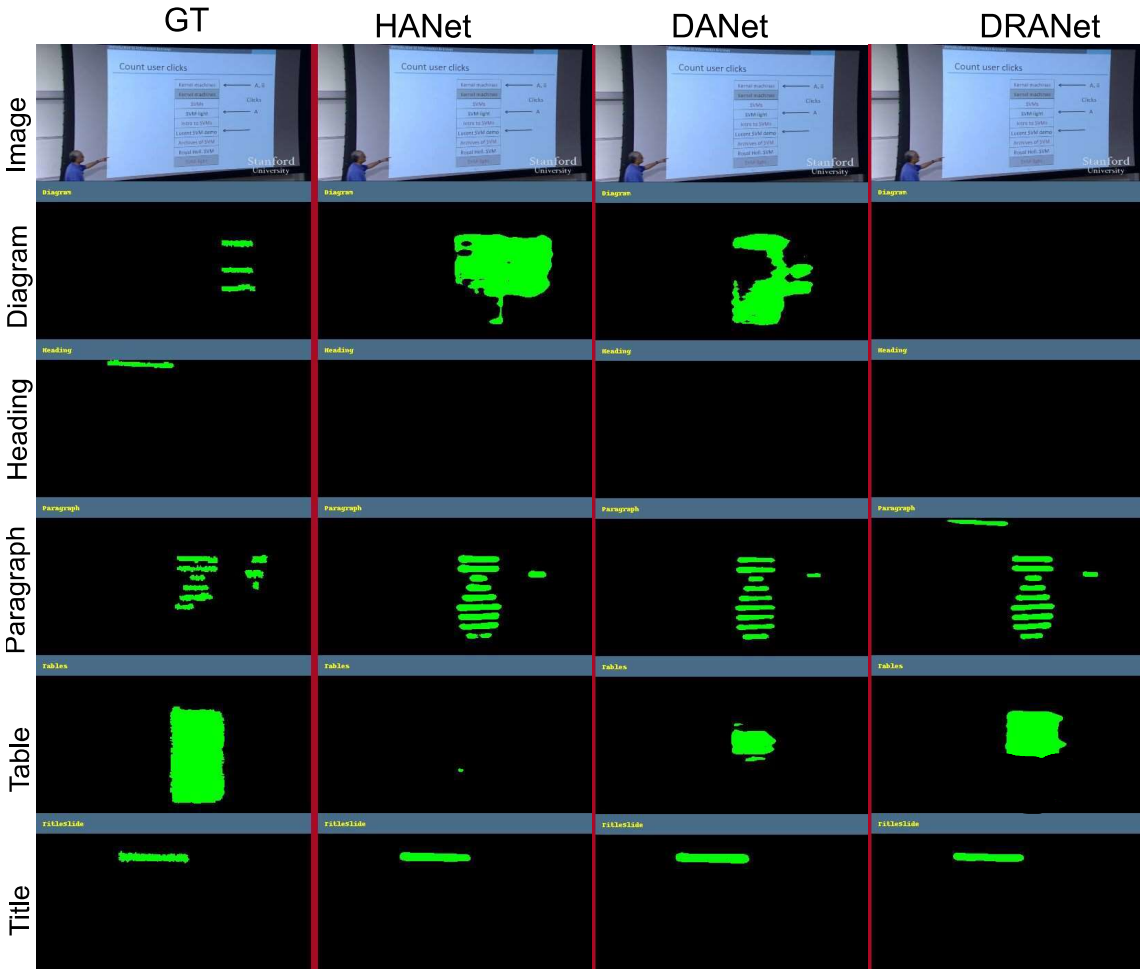


Figure 1.1 The performance of existing image segmentation approaches HANet [9], DANet [10], DRANet [11] on a sample classroom slide images from the WiSe [7] dataset. The results shows that approaches fails to segment table, diagram, and heading regions.

document regions. These approaches involve no post-processing except for a standard non-maximum suppression [25]. This approach fails on documents having non-manhattan layout 1.2(a, c, e).

To overcome this, researchers propose an image segmentation-based approach that classifies each pixel of an image as specific, pre-defined layout categories. The researchers utilize segmentation methods like Fully Convolutional Networks (FCN) to segment documents [26], improving upon object detection models, especially when the logical regions are misaligned, distorted or have complex boundaries. In the problem of historical document image segmentation, the researchers utilize 2D U-Net architecture for document image layout segmentation [27]. Deng *et al.,* [28] proposed an image segmentation-based approach, PixelLink, to segment text instances from scene text images. Text bounding boxes are then extracted directly from the segmentation result without location regression. In the problem of classroom

slide image segmentation, Haurilet *et al.,* [7, 8] proposed a pixel-wise image segmentation based on DeepLab [29].

Further, the attention module [30] enhanced the segmentation accuracy by incorporating it into the segmentation architecture. Fu *et al.,* [10] proposed a self-attention mechanism that adaptively integrates local features with global dependencies. Choi et al., [9] proposed a height-driven attention network (HANet) for improving semantic segmentation for urban-scene images. Fu *et al.,* [11] proposed a Dual Relation-aware Attention Network (DRANet) to handle the task of scene segmentation by capturing contextual information based on the relation-aware attention mechanism. Figure 1.1 shows the qualitative results of classroom slide image segmentation with these attention-based segmentation architectures. Even though these approaches are designed for scene segmentation, they perform well in slide images. However, the approach fails to segment the 'Heading' region, which occurs on top of the slide image and fails to identify the tables as diagram regions. These limitations motivate us to design an attention-based document image segmentation architecture that can incorporate horizontal and vertical positional information.

After successfully segmenting document images, each logical region needs a unique tool to decipher the information. OCR is a tool that decodes text information from paragraphs, tiles, and captions. Understanding the graphical component is also crucial for document understanding. The first step in decoding the graphical region is identifying the graphic type. That required an enriched dataset to train a classification network. Table 1.1 shows the statistics of the existing figure dataset. Most datasets have a limited number of figure categories and several images. These limitations motivate us to create a large dataset containing most labels in scientific documents.

Dataset	**No. of Labels**	**Total Images**
Zhou and Tan [31]	3	1190
Huang and Tan [32]	4	200
Deepchart [33]	5	5000
Figureseer [34]	5	30600
Prasad *et al.* [35]	5	653
Karthikeyani *et al.* [36]	8	155
Revision [37]	10	2000

Table 1.1 Statistics of the existing datasets for document figure classification task.

Finally, let us analyze how the current system explains the document to a blind or visually impaired (VI) student. The output of the two approaches for a given set of slide images is described in Table 1.2. The approach Alt text tried to explain the scene but did not focus on the slide image, and it also wrongly identified the slide as a cell phone. Tesseract OCR is an open-source approach to extract the text in the image. We noticed that no approaches explain a slide image to a VI student. Describing a slide image requires a sophisticated document understanding system. The system should identify the logical regions such as heading, caption, equation, and figure types. The unavailability of such a system motivates

us to design a classroom slide narration system. However, a large language model (LLM) based slide narration systems [38, 39] are performing more accurately.

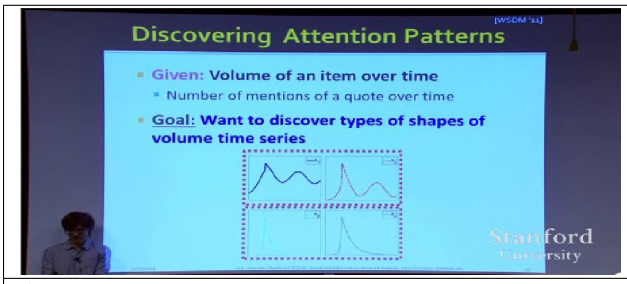
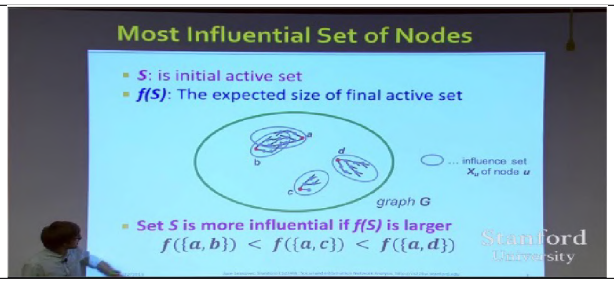
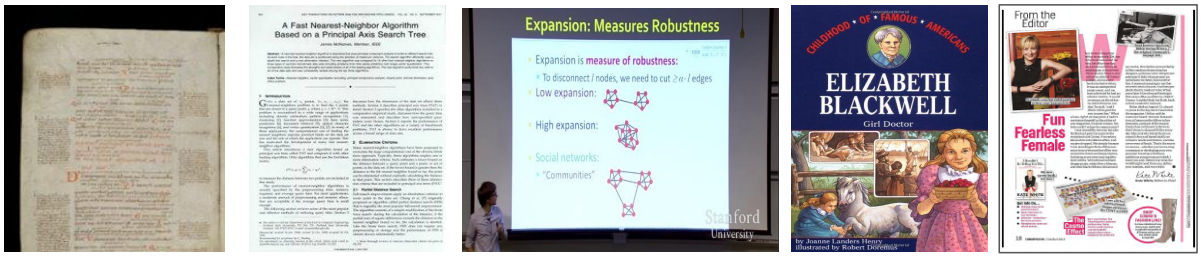
	
Alt text: A screenshot of a person	**Alt text:** A screenshot of a cell phone
Tesseract OCR: Volume of an item over time tions of a quote over time	**Tesseract OCR:**

Table 1.2 The performance of approaches on describing a slide image to a VI student.

These research areas and their associated technologies can help overcome some of the challenges associated with manual analysis, although they still need their own set of considerations and potential limitations. First, the variability in document layout poses a significant challenge. Documents come in diverse formats, from multi-column reports to free-form text, and adapting analysis systems to these varied layouts can be complex. Second, the ambiguity in context adds a layer of intricacy. Document analysis may struggle to discern nuanced meanings or references within the text, especially when relying solely on patterns and structure. Third, the limited semantic understanding of document image analysis systems restricts their ability to comprehend context deeply. Fourth, adapting these systems to different document types, illustrated in Figure 1.2 can be time-consuming, requiring customized training for optimal performance. Lastly, the lack of structural understanding hampers their ability to decipher complex hierarchical relationships within documents, limiting their suitability for more intricate document analysis tasks. Despite these limitations, ongoing research aims to enhance the capabilities of document image analysis systems across a spectrum of applications.



(a) (b) (c) (d) (e)

Figure 1.2 Sample document images having various layouts. (a) historical handwritten document image from the dataset CSG18 [40], (b) scientific document image from PubLayNet dataset [41], classroom slide image from WiSe [7] dataset, (d) book cover image from book cover dataset [42], and (e) magazine image from PRImA layout analysis dataset [43].

1.1 Problems of interest

In the context of the growing interconnection between automatic document image analysis and its practical applications, numerous issues present researchers with challenging inquiries. This section will delve into some intriguing problems within this field.

1.1.1 Deep features in document images

Deep features encompass advanced image representations acquired through deep learning models, primarily focusing on deep neural networks. These representations emerge from processing raw images through multiple layers of neural network architectures, such as convolutional neural networks (CNNs) applied to image data or recurrent neural networks (RNNs) for sequential data. Deep learning models can autonomously extract pertinent and informative features from input images, progressively constructing a hierarchy of increasingly intricate and nuanced representations. These representations encapsulate diverse facets of the data, rendering them particularly conducive for tasks like classification, object detection, image recognition, and natural language processing.

Typically situated within the intermediary layers of a deep neural network, deep features are the product of neurons capturing patterns, edges, textures, and other discerning attributes from the input data. Subsequently, these features serve as inputs for subsequent layers for the network's ultimate output. The salient advantage of deep features lies in their capacity to delineate image data in a manner that expedites intricate pattern recognition and abstraction, often culminating in superior performance across various machine-learning tasks. Researchers frequently harness pre-trained deep learning models to extract these features for utilization in alternative applications, employing a methodology known as transfer learning.

Features are pivotal in document image analysis tasks [44–47], particularly segmentation and classification. Researchers proposed diverse features to address these classification tasks in the context of document image classification [44, 45, 48–50], document figure [46], book cover classification [42], and script identification [47, 51, 52]. However, these approaches are often tailored to specific problems and may need to be more adaptable to others. A feature optimized for a particular task may not perform optimally in a different context. Therefore, the intriguing challenge lies in designing a versatile deep feature network architecture capable of addressing a spectrum of problems.

1.1.2 Document Image Segmentation

Document image segmentation is a crucial process in document image analysis, aimed at partitioning a document image into distinct regions or components. These regions may include text, images, tables, graphics, and other content elements. The primary goal of segmentation is to identify and isolate these components, allowing for more accurate analysis and understanding of the document's structure and content. Document image segmentation techniques involve the detection of boundaries, such as lines or edges, that separate different regions. This process plays a pivotal role in tasks like optical

character recognition (OCR), where text regions need to be separated from other graphical elements, or in document layout analysis, which aims to understand the spatial arrangement of content within a document. Accurate document image segmentation is fundamental for automating document processing tasks and enhancing information retrieval and analysis in various domains, including finance, healthcare, and education systems. In this thesis, we use pre-existing image segmentation models to investigate the difficulties associated with segmenting document images, particularly those from historical, scientific, and lecture slide contexts. Furthermore, we introduce innovative approaches to enhance the quality of segmentation outcomes.

Historic document image segmentation is a specialized field within document image analysis that focuses on strategically segmenting documents with historical significance. These documents often present unique challenges due to their age, degradation, and diverse content. Unlike modern documents, historical documents may exhibit variations in handwriting styles, fonts, and evolved layout conventions. Consequently, document image segmentation demands robust techniques to handle such variability in this context. Researchers and preservationists in the field aim to extract and segment text, decoration, comments, and other elements from historical documents to facilitate their digitization, preservation, and accessibility. Successful historical document image segmentation contributes significantly to preserving cultural heritage, enabling the digitization of valuable historical records, manuscripts, and texts for future generations to study and appreciate.

Scientific document image segmentation is a specialized area within the broader field of document image analysis, focusing on the precise partitioning of documents with scientific content. These documents often include complex elements such as text, equations, tables, figures, and graphs, presenting unique challenges for segmentation algorithms. Accurate segmentation is crucial for various scientific applications, including digitizing research papers, extracting valuable information from scholarly documents, and enabling content-based searches within the scientific literature. Researchers in this field strive to develop sophisticated techniques that can automatically delineate and classify these diverse elements within scientific documents, ultimately enhancing the accessibility and usability of scientific knowledge.

The segmentation of classroom lecture presentation images is a specialized area of document image analysis that focuses on the segmentation of visual content presented during educational lectures. These presentations often contain a mix of text, images, diagrams, and slides with diverse layouts. The segmentation process aims to isolate and categorize these different components, making extracting valuable information easier, assisting in content summarization, or facilitating accessibility for students and educators. Effective lecture presentation image segmentation can enhance the understanding of complex subjects, aid in creating study materials, and contribute to developing educational technologies designed to improve the learning experience in classrooms and online learning environments.

1.1.3 Scientific Document Figure Classification

Figures in scientific documents serve as crucial visual aids that complement textual descriptions and explanations. These graphical representations encompass a wide range of content, including charts, graphs, illustrations, photographs, and diagrams, all carefully selected to concisely convey complex data, patterns, or concepts. Figures play an essential role in enhancing the clarity and comprehensibility of scientific research, enabling researchers to present their findings visually and aiding readers in grasping key insights swiftly. They also serve as a universal language in the scientific community, transcending linguistic barriers to communicate ideas effectively. Furthermore, figures facilitate the reproducibility and validation of research by providing the means to visualize experimental setups, results, and observations. As a result, figures hold a central place in the documentation and dissemination of scientific knowledge, playing a pivotal role in the research and academic publishing process.

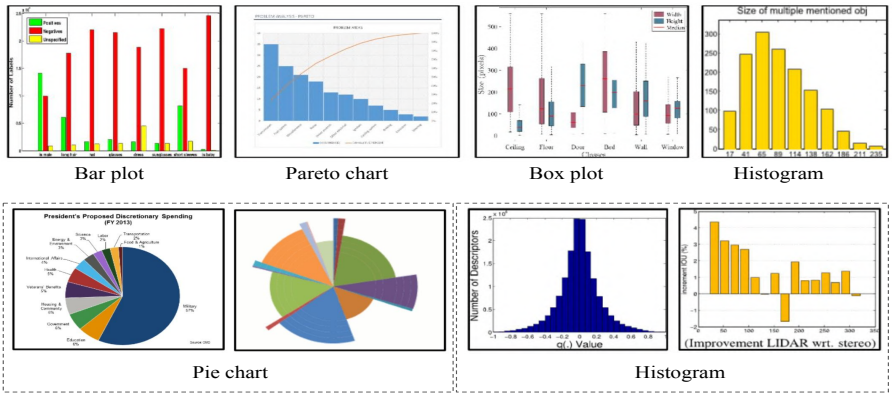


Figure 1.3 The first-row indicates images from different categories share similar visual properties while the second-row highlights that the sample images from a specific type, like *Pie chart* and *Histogram* have a distinct structural properties.

The classification of figures within scientific documents poses several intricate challenges. One of these challenges is distinguishing intra-similarity and inter-dissimilarity among figures, which involves distinguishing between figures that share similarities within a figure class and those that differ significantly. Overcoming this challenge requires sophisticated feature extraction and pattern recognition techniques. Another hurdle is the need for comprehensive datasets for training deep neural networks specifically for figure classification tasks. Gathering a diverse and representative dataset can be particularly demanding due to the vast array of scientific disciplines and document types. Additionally, factors like figure style variations, resolution, and format further complicate the classification process. Overcoming these challenges is paramount for enhancing the efficiency and accuracy of figure classification in scientific documents, ultimately aiding researchers and educators in accessing and utilizing scientific knowledge effectively.

Limited work on document figure (mainly various types of charts) classification has been done in the literature [31, 33–35, 37, 53, 54]. The existing methods [31, 35, 37, 53, 54] based on handcrafted features fail to achieve good accuracy on classification of figures in the document images due to large visual similarity among subordinate categories. The existing datasets (e.g. FigureSeer [34], Revision [37], Deepchart [33], Karthikeyani and Nagarajan [36], Prasad *et al.* [35], Huang and Tan [32], Zhou and Tan [31]) on the classification of document figures (mainly charts) are limited with respect to both the samples (less than or equal to 5K except FigureSeer) and category labels (less than or equal to 10).

1.1.4 Applications of Document Image Layout Segmentation

Document image layout segmentation has applications encompassing a broad spectrum of fields and industries. In document analysis and management, layout segmentation is instrumental in automating tasks such as document categorization, indexing, and retrieval, significantly streamlining information organization and access. In the publishing and media industries, it plays a pivotal role in formatting and structuring content for both print and digital platforms, ensuring optimal readability and aesthetics. In the context of historical document preservation, layout segmentation aids in the digitization and preservation of valuable manuscripts, making them accessible to a broader audience. Moreover, it is integral to optical character recognition (OCR) systems, enabling accurate document text recognition. In the medical field, layout segmentation supports extracting relevant information from medical records, contributing to efficient healthcare administration. These applications underscore the versatility and importance of document image layout segmentation in enhancing data processing, retrieval, and information management across various sectors.

In the realm of education, the applications of document image layout segmentation are diverse and impactful. One notable application is digitizing educational materials, including textbooks, lecture notes, and research papers. Layout segmentation helps convert these materials into digital formats, making them easily accessible to students and educators through online platforms, e-learning systems, and digital libraries. It also plays a crucial role in creating interactive e-learning content, allowing for precisely identifying and extracting text, images, diagrams, and other educational elements. Moreover, layout segmentation aids in developing content summarization tools, enabling students to grasp the key concepts quickly and the main points of complex documents. It also contributes to the customization of educational materials, allowing educators to adapt content for different teaching approaches and student needs. In essence, document image layout segmentation enhances the digitization, accessibility, and effectiveness of educational resources, fostering a more dynamic and interactive learning experience.

In our thesis, we place particular emphasis on the development of several innovative applications with diverse and impactful purposes. Firstly, we delve into scientific document layout transferring, a valuable tool that empowers readers to select an optimal layout tailored to their preferred device, be it a mobile device, tablet, or computer screen. This adaptation enhances the reading experience, ensuring content accessibility across various platforms. Additionally, we focus on a Slide Narration System designed to address the needs of visually impaired (VI) students who often face challenges in comprehending

classroom slide presentations. With the help of a mobile phone, this system assists VI students in accessing and interpreting slide content through narration. Lastly, we explore a Slide Image Retrieval System, which is relevant in the Massive Open Online Courses (MOOCs) era. With the proliferation of educational materials, especially lecture slides, on the web, this system has become instrumental in facilitating efficient content retrieval, enabling users to search and access relevant materials amid the vast sea of information.

1.2 Contributions

1.2.1 Deep feature for Historic document image segmentation

We explores the effectiveness of deep features for document image segmentation. The document image segmentation problem is modelled as a pixel labeling task where each pixel in the document image is classified into one of the predefined labels such as text, comments, decorations and background. Our method first extracts deep features from superpixels of the document image. Superpixels are compact, non-overlapping regions or groups of pixels within an image that share similar color, texture, or intensity characteristics. They are typically generated through an over-segmentation process that divides an image into homogeneous regions based on low-level features. Unlike individual pixels, superpixels are larger and represent coherent image regions, making them a useful abstraction for various computer vision tasks.

We extract deep features from superpixels, employing Fisher vector encoded convolutional layers (FV-CNN) and fully connected layers (FC-CNN). These features encapsulate rich superpixel information, paving the way for precise classification. Subsequently, we exploit the power of a Support Vector Machine (SVM) classifier to predict the class for each superpixel, effectively categorizing them. Finally, generate a segmentation output that accurately delineates the different layout regions within the document image. Rigorous experimentation and evaluation validate the effectiveness of our method, demonstrating its superiority over popular approaches when applied to benchmark handwritten datasets.

1.2.2 Design a deep feature extraction network for various document images

In document image analysis, salient features such as texture or repeating patterns, discriminative patches, and shapes play a pivotal role in addressing diverse challenges. For instance, when classifying languages of scripts at the word level, texture features prove invaluable, while layout-level shape features excel in distinguishing various document types. In response to these multifaceted demands, we propose a novel deep network architecture that autonomously learns these critical features, encompassing texture patterns, discriminative patches, and shapes. We design an approach to tackle various document image analysis tasks, including document image classification, genre identification from book covers, scientific document figure classification, and script identification.

The network we present is adept at learning global, texture and discriminative features independently and adeptly combines them according to the specific nature of the problem at hand. In this study, we have devised three distinct categories of deep features, each tailored to capture information from different scales within the image. The global feature encompasses the shapes and overall structures inherent in the image. Conversely, the texture feature captures the finer details, such as fonts or script types employed within the image. Lastly, the discriminative feature identifies the distinctive regions unique to each class. We evaluate the performance of our approach against state-of-the-art techniques using multiple publicly available datasets, including Book-Cover, RVL-CDIP, CVSI, and `DocFigure`. Diverse experiments conclude that a combination of discriminative, texture, and global features performs better for various classification tasks - document image classification, genre classification of book, document figure classification, and script identification. The experimental results showcase the efficacy of our approach, particularly in the realms of genre and document figure classifications, where it surpasses the state-of-the-art while also demonstrating competitive results in document image and script classification tasks.

1.2.3 Enhancing slide image segmentation using location embedding

Slides play a pivotal role as essential document types utilized as visual aids in various domains, encompassing education and business. Over time, their significance has evolved from mere supplements to spoken presentations or printed materials to becoming the primary medium for disseminating knowledge, especially within university classrooms. Slides exhibit a heightened emphasis on visual content and manifest considerable diversity in their structural composition, layouts, and the relationships between entities presented, often culminating in intricate designs. Analyzing unconstrained photographs of slides captured during live lectures poses a formidable challenge in the domain of document image analysis. While numerous research endeavors have tackled page segmentation issues in conventional documents, exploring slide analysis needs to be more extensive, earning further investigation and innovation in this evolving field.

This problem poses as extracting logical regions such as title, text, equation, figure, and table from the slide image. The goal is to assign logical labels like title, paragraph, list, equation, table, and figure to each pixel in the slide images. Due to the large variability in theme (i.e., the layout), style, and slide content, extracting meaningful regions becomes a challenging task. In the classroom slide images, the logical regions are distributed based on the location of the image. We propose a location-encoded attention module that utilizes the location encoding of logical regions of slide images. To utilize the location of the logical regions for slide image segmentation, we propose the architecture, Classroom Slide Segmentation Network (CSSN). To enhance the segmentation accuracy, we concatenate the attention module output feature with the multi-scale features through an Atrous Spatial Pyramid Pooling (aspp) layer [29]. The unique attributes of this architecture differ from most other semantic segmentation networks. Publicly available benchmark datasets such as WiSe [7] and SPaSe [8] are used to validate the performance of our segmentation architecture.

1.2.4 Dataset for figure classification and slide image retrieval

Automatic Document Figure Understanding (DFU) represents a pivotal technological advancement with far-reaching implications across diverse applications. It is vital in enhancing accessibility, enabling efficient information retrieval, supporting knowledge extraction, facilitating data analysis, and streamlining content generation processes in various domains.

In the context of DFU, Document Figure Classification (DFC) emerges as a crucial component, as it forms the initial step of identifying the figure category for understanding the visual content within the figure. However, the design of an effective DFC system hinges on the availability of well-defined figure categories and robust datasets. To train modern neural networks, the available datasets for classifying figures in document images could be more extensive in size and category diversity.

In response to this limitation, we introduce the `DocFigure` dataset, a comprehensive collection comprising 33,000 annotated figures categorized into 28 distinct classes. We collected the figures from scientific articles published in esteemed conferences such as CVPR, ECCV, ICCV, and others over recent years. Given the impracticality of manually annotating such a vast dataset, we have developed an efficient web-based annotation tool to streamline the assignment of category labels with minimal human annotator effort. To benchmark the utility of our curated dataset for classification tasks, we propose three baseline classification techniques: leveraging deep features, deep texture features, and a combination of both. Our comprehensive approach aims to facilitate further advancements in DFC and DFU, fostering innovation in document understanding and accessibility.

Massive Open Online Courses (MOOCs) have ushered in a new era of educational accessibility, granting individuals easy and open access to a wealth of educational resources, especially lecture slides, through online platforms. However, the sheer volume of available materials poses a formidable challenge when searching and retrieving specific content based on user queries. In response to this challenge, recent years have witnessed a surge of interest among researchers in developing advanced AI systems [7, 8, 55–58]. These systems harness the power of educational lecture videos and presentation slide images to revolutionize the educational landscape, offering innovative solutions for content organization, retrieval, and personalization. The synergy between AI technology and educational resources holds the promise of enhancing learning experiences, making educational materials more accessible, and streamlining the process of knowledge acquisition and dissemination. Multiple AI systems can easily design new slides by combining search results and reusing figures and graphs from existing slide images, minimizing manual effort.

Existing recommendation and retrieval systems [55, 59] have primarily been engineered to retrieve slides from video files but need help with several inherent limitations. Firstly, these systems heavily rely on the transcript associated with the video content to facilitate slide retrieval. However, this approach often falls short in terms of contextual alignment with the slide image, and the transcript may not always be accessible or available, mainly when dealing with collections of lecture slide images rather than videos. Secondly, these retrieval systems are constrained because they exclusively accommodate text-

based queries, effectively excluding the valuable dimension of hand-drawn sketches or diagrams as valid search queries.

Furthermore, text-based queries often encompass logical regions within slides, such as titles, bullet points, and figures, along with many figure types, including line graphs, bar charts, Venn diagrams, and tree diagrams. Hence, a clear imperative arises for developing a specialized dataset is required. This dataset must accommodate user-defined queries, encompassing textual and sketch-based inputs, bridging the gap in current retrieval systems and facilitating innovation in document image analysis. We introduce a novel dataset, namely the Lecture Slide Deck (LecSD) Dataset containing $54K$ slide images from the Data Structure, computer networks, and optimization courses and provide associated manual annotation for the query in the form of natural language or hand-drawn sketch.

1.2.5 Applications

The ultimate aspiration is to translate our research into practical solutions that address real-world challenges and enhance the quality of life for individuals and communities. In line with this vision, we have also conceptualized and designed a range of innovative applications that leverage the fruits of our research. These applications represent a concerted effort to harness the transformative potential of our work and make meaningful contributions to various domains, bridging the gap between theory and impactful, real-world solutions.

1.2.5.1 Classroom Slide Narration System

Slide presentations represent a prevalent and efficient pedagogical tool educators employ for effective classroom communication. Nonetheless, this conventional teaching model presents significant challenges for students who are blind or visually impaired (VI). These students often necessitate personal human assistance to comprehend the visual content presented in slides, posing barriers to independent learning. Recognizing this limitation, we were inspired to develop a transformative solution known as the Classroom Slide Narration System (CSNS). The core objective of CSNS is to empower VI students by generating audio descriptions that seamlessly align with the content depicted in the slides. By providing accessible and tailored audio descriptions, CSNS aims to bridge the educational divide and enhance the learning experience for all students, ensuring inclusivity and equity in the classroom.

1.2.5.2 Document Image Reformatting

In the contemporary digital landscape, smartphones have emerged as powerful tools for document reading, thanks to significant advancements in digital media processing. However, despite their convenience, the visualization of double-column formatted documents on the relatively small screens of smartphones poses a unique challenge for readers. Efforts to address this issue have included reflowing documents to suit the constraints of mobile portable devices, garnering notable attention in recent years. Nevertheless, these solutions primarily cater to text-based documents and face limitations when

confronted with image-based documents. Our research endeavors delve into document reformatting, which we define as transforming input document images with one layout style into target document images characterized by a different layout style. By tackling this intricate problem, we seek to allow users to seamlessly adapt and access a diverse range of document content, transcending the constraints of varying layout styles on digital platforms.

1.2.5.3 Lecture Slide Deck Search Engine

Massive Open Online Courses (MOOCs) have significantly democratized education by offering easy access to a wealth of educational resources, particularly lecture slides, via the Internet. However, as the volume of available content continues to burgeon, effectively searching and retrieving specific information becomes paramount. To address this crucial challenge, we introduce the Lecture Slide Deck Search Engine (LecDeckSearch Engine), a pioneering search platform to facilitate seamless information retrieval. What sets LecDeckSearch Engine apart is its versatility, as it supports both natural language queries and hand-drawn sketches, enabling users to search effortlessly within a vast repository of slide images covering computer science topics. This innovative search engine supports a novel semantic label-aware transformer model, which excels at extracting and encoding semantic labels within slide images. By seamlessly fusing these semantic cues with visual information from the slide images and textual cues from natural language queries, LecDeckSearch Engine empowers users to navigate the ocean of educational content with precision and ease.

1.3 Thesis Overview

The rest of the thesis is organized in the following parts.

Background and Related Work: This section serves as an intellectual foundation, delving into the fundamental concepts, theorems, models and a comprehensive exploration of prior research that form the bedrock of our thesis. By unraveling these essential components, we aim to provide readers with a comprehensive understanding of the theoretical underpinnings and the broader intellectual landscape within our research. This contextualization lays the groundwork for the subsequent sections of our thesis, where we delve into the practical applications, methodologies, and findings that contribute to the broader academic discourse.

Deep Multi-modular Features: Texture patterns, discriminative patches, and shapes constitute pivotal salient features that underpin a broad spectrum of document image analysis challenges. In the context of this chapter, we unveil a novel deep network architecture meticulously crafted to autonomously acquire proficiency in discerning these crucial features. This architectural innovation forms the cornerstone of our approach, enabling us to address a diverse array of document image analysis tasks with precision and efficacy. The tasks we undertake include document image classification, genre identification based on book covers, scientific document figure classification, and script identification. Our presented network adopts a versatile learning approach, autonomously assimilating global, texture, and

discriminative features. Subsequently, it tactically combines these features, judiciously aligning them with the unique demands of each problem at hand. Through this comprehensive framework, we endeavor to demonstrate the potential of deep learning in advancing the field of document image analysis and expanding its applications across various domains. The chapter includes our published work in the Journal of SN Computer Science in 2022.

Document Image Layout Segmentation: This chapter comprehensively explores our significant contributions within the multifaceted domain of document image segmentation. We have rigorously tackled the research task across three distinct document image categories: a) historical documents, b) scientific materials, and c) images captured during classroom lecture presentations. This chapter mainly discuss our work published in Computer Vision, Pattern Recognition, Image Processing and Graphics (NCVPRIPG) in 2017 and Computer Vision and Image Processing (This chapter mainly discuss our work published in Computer Vision, Pattern Recognition, Image Processing and Graphics (NCVPRIPG) in 2017 and Computer vision and image processing in 2021. (CVIP) in 2021.

Dataset for document image analysis In our research journey, we contribute two noteworthy datasets to the document image community, marking significant milestones in our endeavor. The first of these, the `DocFigure` dataset, encompasses a rich repository comprising 33,000 images thoughtfully categorized into 28 diverse figure classes. This dataset is a valuable resource for researchers, facilitating advancements in various domains of document image analysis. In parallel, we introduce the LecSD (Lecture Slide Deck) dataset, specifically tailored to support slide image retrieval. LecSD enables users to employ textual queries and hand-drawn sketches as retrieval inputs, offering a versatile and comprehensive platform for the research community. These are from our published work in the IAPR International Workshop on Graphics Recognition 2019.

Applications based on document image analysis This chapter explains an in-depth exploration of the practical applications of our research in document image analysis. We pride ourselves on presenting three innovative applications, each meticulously designed to address distinct real-world challenges. First among them is the Classroom Slide Narration System (CSNS), a pioneering solution aimed at enhancing accessibility and learning for individuals who are blind or visually impaired. CSNS generates audio descriptions corresponding to the content of classroom slides, empowering visually challenged students with independent access to educational materials. Additionally, we delve into document reformatting, a transformative endeavor that adapts document images from one layout style to another. This application bridges the gap between different layout styles, ensuring seamless access to document content across diverse digital platforms. Finally, we unveil the Slide Deck Search Engine (LecDeckSearch Engine). This versatile platform supports natural language and hand-drawn sketch queries for efficient searches across a vast collection of slide images, catering to the information retrieval needs of learners and educators alike. These applications collectively epitomize our commitment to harnessing the power of document image analysis for practical, real-world solutions, advancing accessibility, and enriching the learning experience.

Conclusion Finally, we conclude the thesis by summarizing the contributions and drawing directions for future research in related areas.

Chapter 2

Background and Related Work

Document image analysis, a specialized branch of computer vision and image processing, centers on extracting and examining textual and other content found in scanned, camera-captured, or digital documents. In this chapter, we explore the background and the classical approaches that researchers have proposed for the problems introduced in Chapter 1. Researchers employ a variety of computer vision algorithms for document image analysis, often introducing innovative algorithms tailored to this specific field. This chapter provides an overview of the fundamental algorithms utilized in document image analysis, highlighting prior approaches that serve as the foundation for our contributions in this thesis.

Automatic document image analysis has a rich history spanning over six decades. We divide this extensive timeline into two distinct epochs: the pre-deep learning era and the era of deep learning approaches. Classic image representations and document image segmentations were the prevailing methodologies during the pre-deep learning era. In contrast, the advent of deep learning approaches has led to remarkable advancements, consistently surpassing the performance of classical methods across a spectrum of document image analysis tasks. Nevertheless, researchers have adopted a hybrid approach, amalgamating classical techniques with deep learning models to propose cutting-edge solutions in computer vision and document image analysis domains.

2.1 Document image analysis in Pre-deep learning era

In this section, we elucidate the traditional image representation that demonstrated effectiveness in the era preceding the widespread adoption of deep learning. In computer vision and image processing, feature extraction is pivotal in transforming visual data into a more manageable and informative representation. This process entails mapping an image, denoted as $I \in \mathcal{I}$, into a feature space$X \in \mathcal{X}$ or learning a mapping function$\mathcal{F} : \mathbb{R}^p \rightarrow \mathbb{R}^d$. Here, p signifies the number of pixels within the image, and d represents the feature dimension, often characterized by $d << p$. Selecting an appropriate feature mapping function hinges on the specific task under consideration. Ideally, such a function should exhibit several critical properties, including (i) invariance to translation, scale, and illumination, ensuring the features remain robust under varying conditions; (ii) resilience to noise and degradation, preserving

feature integrity in the presence of disturbances; (iii) compactness, resulting in a concise representation of the underlying information; and (iv) computational efficiency, facilitating swift and efficient feature computation. These properties collectively guide the design and implementation of feature extraction methods in the pursuit of more effective and robust computer vision systems. Various techniques and methods, including the following, can achieve feature extraction:

- **Interest Point Detection:** They identify key points or interest points in an image representing unique regions, such as corners, edges, or blobs. For this purpose, researchers use algorithms such as Harris corners [60], Shi-Tomasi corners [61], and FAST [62]. Corners are valuable because they represent regions where image intensity or color changes significantly in multiple directions. They are helpful for image matching, tracking, and feature-based object recognition.

  These approaches employed in document image analysis have garnered significant attention due to their effectiveness in addressing complex tasks. Writer identification, for instance, has seen notable advancements through integrating deep learning techniques in combination with fast key points and the Harris corner detector, as evidenced by the work of Semma *et al.* [63]. Furthermore, skew detection and correction techniques have harnessed the Hough transform and Harris corner detector's power, as Gari *et al.* illustrated [64]. At the same time, document page layout analysis has hinged on using Harris corner points, as described by Nourbakhsh *et al.* [65]. Additionally, skew correction methodologies for vehicle license plates have effectively harnessed the principal component of Harris corner features, as detailed in the work of Modi *et al.* [66]. These approaches collectively contribute to the robust and comprehensive field of document image analysis, offering valuable insights and solutions to various challenges in this domain.

- **Local Descriptors:** Researchers have developed numerous feature extraction techniques to capture distinctive image patterns and information. Four noteworthy methods that have made substantial contributions in this domain are Scale-Invariant Feature Transform (SIFT) [67], Speeded-Up Robust Features (SURF) [68], Oriented FAST and Rotated BRIEF (ORB) [69], and Binary Robust Independent Elementary Features (BRIEF) [70]. SIFT, introduced by David Lowe in 1999, is renowned for its scale and rotation invariance, making it particularly effective for object recognition. SURF, proposed by Herbert Bay *et al.* in 2006, offers similar robustness and excels in computational efficiency. ORB, devised by Ethan Rublee *et al.* in 2011, combines speed with robustness, making it suitable for real-time applications. BRIEF, introduced by Michael Calonder *et al*. in 2010, is efficient in creating binary descriptors. These feature extraction techniques have found diverse applications across various computer vision tasks, including object recognition, image matching, and image retrieval.

  The utilization of these approaches has demonstrated notable success in addressing diverse document analysis tasks, showcasing their versatility and effectiveness. For instance, identifying scripts from handwritten documents has benefited from implementing the SIFT method, as illustrated in the work by Rajput *et al.*. [71]. Similarly, Farsi and Arabic optical font recognition have

leveraged the discriminative power of SIFT features, as exemplified in the study conducted by Zahedi *et al.* [72]. Furthermore, Ahmed *et al.* [73] have applied the SURF method to facilitate the extraction of text embedded within graphical elements. Additionally, the domain of printed document forensics has witnessed advancements through the computational approach, incorporating SURF and ORB features, as proposed by Kumar *et al.* [74]. These endeavors underscore the practical utility of these approaches in addressing various challenges encountered in document analysis, thus contributing significantly to the advancement of this field.

- **Histogram-based Features:** Histogram-based feature extraction methods in computer vision involve analyzing the distribution of pixel values, colors, or gradients within an image to capture valuable information about its content. These methods represent image characteristics in histograms, which are histograms of pixel values, color intensities, gradients [75], and texture [76]. Histogram-based features are valuable for various computer vision tasks, including image classification, object recognition, and image retrieval.

  Histogram-based Features play a pivotal role in the analysis of document images, and their effectiveness is evident in several noteworthy studies. For instance, Cruz *et al.* utilized Local Binary Patterns (LBPs) to detect document forgery, as documented in their work [77]. Similarly, Nicolaou *et al.* introduced Oriented Local Binary Patterns in the realm of writer identification, demonstrating their relevance in this context [78]. Furthermore, Dey *et al.* applied Local Binary Patterns for word spotting in handwritten historical documents, showcasing the versatility of this approach [79]. In scene text recognition, Tian *et al.* harnessed the Co-occurrence of Histogram of Oriented Gradients (HOG) to achieve notable results [80]. Additionally, Tian *et al.* extended the utility of HOG-based features to multilingual scene character recognition, emphasizing the adaptability of these techniques [81]. These studies underscore the significance of Histogram-based Features in advancing various aspects of document image analysis and scene text recognition.

In this thesis, our initial focus is on elucidating the fundamental concept of Bag of Words (BoW), often called bag-of-visual-words (BoVW), in visual data analysis. Seminal works, such as [82, 83], trace the inception of this concept. The BoW model, along with its subsequent advancements, adeptly amalgamates the capabilities of local descriptors and histogram-based approaches, imparting efficiency to various aspects of visual data analysis. In the ensuing section of this report, we delve deeper into the Bag of Words concept and explore its successor methodologies, thus offering a comprehensive understanding of their transformative role in visual information processing.

2.1.1 Bag of Visual Words (BoVW) Representation

The foundational premise of the Bag of Visual Words (BOVW) model operates under the fundamental assumption that an image can be conceptualized as an amalgamation of visual words, drawing parallels to the relationship between textual documents and their constituent words. These visual words, denoted as $b \in B$, form an integral part of a visual vocabulary denoted as B, which is acquired through

unsupervised learning from a subset of the data, typically a collection of images. When presented with an image, an essential BOVW representation is constructed by mapping each local descriptor to its nearest visual word and aggregating them into a histogram of visual words. The BOVW framework follows a typical pipeline encompassing various stages, including (i) interest point detection, (ii) computation of local descriptors, (iii) creation of the visual vocabulary, often referred to as quantization, (iv) coding, and (v) pooling.

2.1.1.1 Detectors and Descriptors

In the bag-of-visual-word framework, the primary step involves extracting interest points (keypoints) and then computing the descriptors. Several methods exist for identifying keypoints, including approaches such as the unsupervised discovery of mid-level discriminative patches [84] and exploring distinctive components for scene classification [85]. These interest points can exhibit varying characteristics, being either sparse or dense, and an essential property sought in a detector is repeatability, which enhances its robustness against noise and various forms of degradation.

As a low-level image processing technique, feature detection leverages information related to edges, corners, or blob-level characteristics. While different detectors are typically more suitable for specific domain tasks, subsequent research has revealed that utilizing dense or regularly distributed keypoints at various scales can also yield competitive performance. Several well-known keypoint detectors have gained prominence in the field of computer vision. These include the Harris affine detector, which, as outlined in Harris' work [60], specializes in computing corner points while preserving those with higher response values. Another noteworthy detector is the Difference of Gaussians (DoG), which is an integral part of the SIFT descriptor, as introduced by Lowe [86]. The DoG detector identifies local maxima points within a scale-space created through the difference of Gaussian pyramids. While it is common practice to select keypoint detectors based on specific domain tasks, subsequent research has revealed that employing dense or regularly distributed keypoints at various scales, as evidenced by studies like [87, 88], can yield competitive performance across a range of applications.

2.1.1.2 Visual Vocabulary

Consider a collection of K descriptors, denoted as $x_1, x_2, \ldots, x_K \in \mathbb{R}^d$, obtained from a random subset of the entire image corpus. To illustrate this concept, we take the example of the SIFT descriptor, characterized by a dimensionality of $d = 128$ and recognized as a state-of-the-art descriptor in BOVW-based image retrieval methodologies. The process of learning a visual vocabulary essentially entails partitioning the local descriptor space into meaningful and informative regions. Typically, this is achieved using a k-means based clustering algorithm that minimizes the sum of squared Euclidean distances between sample points and their corresponding cluster centres. Mathematically, this optimization process is represented as:

$$B^* = \underset{B}{\operatorname{argmin}} \sum_{i=1}^{M} \sum_{x \in S_i} \|x - \mu_i\|_2^2 \tag{2.1}$$

Here, $S = \{S_1, S_2, \ldots, S_M\}$ represents the set of M clusters, with each cluster S_i containing descriptors closest to the cluster mean denoted as μ_i. The term $B = [\mu_1, \mu_2, \ldots, \mu_M]$ refers to the learned visual vocabulary, commonly known as the codebook, where each μ_i represents a codeword. This entire process is referred to as quantization. The selection of the codebook size, denoted as M, should strike a balance: it should not be too small to under-represent patches from the corpus, nor too large, which could introduce quantization errors. Quantization errors manifest as ambiguity when assigning codewords to descriptors that are in proximity to the boundaries of two or more clusters. Typically, the value of M is empirically determined through cross-validation. It is important to note that learning a visual vocabulary via k-means on real-world datasets containing millions of descriptors can be computationally demanding.

Researchers have developed efficient clustering methods to tackle this challenge. One example is hierarchical clustering, introduced by Nister et al. [89], and another is approximate k-means using randomized kd-trees, proposed by Philbin et al. [90]. These methods significantly reduce the computational complexity of assigning a sample to a cluster, achieving a complexity of $O(d \log M)$, where 'd' represents the dimensionality of the data and 'M' stands for the codebook size.

2.1.1.3 Coding and Pooling

Moving on to the next stage in the pipeline, we encounter the "coding" process, which accomplishes the mapping from the input feature space $\left(\mathbb{R}^d\right)$ to codewords $\left(\mathbb{R}^M\right)$. The most widely adopted and fundamental coding technique is vector quantization (VQ). In VQ, each descriptor is assigned to its nearest codeword from the codebook, representing a hard quantization approach where each descriptor is associated with precisely one codeword. Mathematically, the VQ operation for an image comprising N descriptors are expressed as follows:

$$\begin{aligned} C^* &= \underset{C}{\operatorname{argmin}} \textstyle\sum_{i=1}^{N} \|x - Bc_i\|^2 \\ \text{s.t. } &\|c_i\|_{l_0} = 1, \|c_i\|_{l_1} = 1, c_i \geq 0, \forall i \end{aligned} \tag{2.2}$$

The code $c_i \in \mathbb{R}^M$ corresponds to the code associated with the i^{th} descriptor. To construct the Bag of Words (BOW) representation from the set of codes $C = \{c_1, c_2, \ldots, c_N\}$ obtained from a given image, a process known as "pooling" is performed. There are two primary types of pooling methods: (i) sum pooling (or average pooling), represented as $h = c_1 + c_2 + \ldots + c_N$, and (ii) max pooling, represented as $h = \max(c_1, c_2, \ldots, c_N)$. Here, the resulting representation denoted as h" holds several advantages, including deriving a compact representation, imparting invariance against variations in position and illumination conditions, and enhancing robustness against clutter, among other benefits. However, it is essential to note that straightforward pooling also leads to a loss of geometrical information, which

represents a fundamental limitation within the bag of words model. This geometrical information is crucial for building robust image retrieval or recognition systems. Lazebnik et al. [91] introduced a spatial pooling approach regain the lost geometric information partially. This method involves dividing an image into multiple spatial pyramids and then combining the weighted histograms generated from individual regions within these pyramids. Specifically, this approach assigns higher weights to regions at finer levels, distinguishing them from regions at coarser levels.

2.1.2 Higher Order Representations

The discussed encoding techniques have primarily concentrated on capturing the count statistics of visual words. From a statistical standpoint, one can liken the Bag of Visual Words (BOVW) representation to capturing the zeroth moment from a set of data samples. However, there is the potential to derive higher-level representations by utilising first and second-order moments. In this context, two widely recognised representations come into play: (i) The Vector of Linearly Aggregated Descriptors (VLAD) method proposed by Jégou et al. [92]., which leverages the information contained in the first moment, and (ii) Fisher Vectors (FV) introduced by Perronnin and Dance [92], which harnesses the statistics of the first, second, and third moments.

2.1.2.1 Vector of Locally Aggregated Descriptors (VLAD)

When considering a codebook denoted as $B = \{b_i, i = 1, \dots M\}$ learned using k-means and a collection of local descriptors represented as $X = \{x_t, t = 1, \dots, T\}$, the VLAD computation involves three distinct stages:

Stage 1: Assignment of Nearest Neighbors - Each local descriptor x_t is assigned to its nearest visual word using the expression $NN\left(x_t\right) = \operatorname{argmin}_{b_i} \|x_t - b_i\|$.

Stage 2: Residual Vector Computation - Residual vectors v_i are computed for each visual word b_i by aggregating the differences between the local descriptors x_t and their assigned visual word, i.e., $v_i = \sum_{x_t : NN(x_t) = b_i} \left(x_t - b_i\right)$.

Stage 3: Concatenation and L_2 Normalization - The final representation is obtained by concatenating the computed residual vectors and then performing L_2 normalization on this concatenated vector.

As illustrated in the above stages, the ultimate representation is a concatenated and L_2 normalized feature obtained from each visual word. In this representation, we characterize each visual word by aggregating the residual vectors representing the first-order moment of all the descriptors in X assigned to that particular visual word. It is important to note that compared to earlier coding schemes, which resulted in a dimensionality of M (where M is the number of visual words), the VLAD representation exhibits increased dimensionality of $d \times M$, with d being the dimensionality of the descriptor.

2.1.2.2 Fisher Vectors (FV)

Fisher Vectors (FVs) represent an application of Fisher Kernels (FK) on visual vocabularies, offering a dense description of an image while utilizing smaller vocabularies. FK serves as a method for measuring the similarity between a data sample and a statistical model, aiming to combine the advantages of both generative and discriminative approaches. For a given sample X and a likelihood function P_λ with parameters λ, the scoring function is expressed as follows:

$$\phi_\lambda(X) = \nabla_\lambda \log p(X \mid \lambda) \tag{2.3}$$

This scoring function signifies the direction in which the model parameters λ should be adjusted to better align with the data, resulting in a fixed-length vectorial representation whose size is determined by the number of parameters. The Fisher Kernel is defined as:

$$K(X, Y) = \phi(X \mid \lambda)' F_\lambda^{-1} \phi(Y \mid \lambda) \tag{2.4}$$

Here, F_λ represents the Fisher information matrix (FIM), which normalizes the input vector and is defined as:

$$F_\lambda = E_X \left[\nabla_\lambda \log p(X \mid \lambda) \nabla_\lambda \log p(X \mid \lambda)' \right] \tag{2.5}$$

Perronnin and Dance [93] introduced Fisher Kernels (FK) for visual vocabularies, representing vocabularies using a Gaussian Mixture Model (GMM). Let $X = \{x_t, t = 1, \dots T\}$ denote a set of T i.i.d. d-dimensional local descriptors from an image, and let λ represent the parameters of the GMM, which includes w_i, μ_i, and Σ_i for Gaussian component i, where w_i is the weight, μ_i is the mean vector, and Σ_i is the covariance matrix. M denotes the total number of Gaussians in the GMM. The scoring function for X is given as follows, reflecting average pooling due to the i.i.d. assumption:

$$\phi_\lambda(X) = \frac{1}{T} \sum_{i=1}^{T} \nabla_\lambda \log p\left(x_t \mid \lambda\right) \tag{2.6}$$

$$\text{where, } p\left(x_t \mid \lambda\right) = \sum_{i=1}^{M} w_i p_i \left(x_t \mid \lambda\right) \tag{2.7}$$

$$\sum_{i=1}^{M} w_i = 1 \tag{2.8}$$

$$p_i(x \mid \lambda) = \frac{\exp\left\{-\frac{1}{2}\left(x - \mu_i\right)' \Sigma_i^{-1} \left(x - \mu_i\right)\right\}}{(2\pi)^{D/2} \left|\Sigma_i\right|^{1/2}} \tag{2.9}$$

The scoring function $\phi_\lambda(X)$ is computed by taking derivatives with respect to each w_i, μ_i, and Σ_i, normalized by the Fisher Information Matrix, and concatenated to obtain the final image representation. This results in a representation size of $(2 \times d + 1) \times M$.

2.1.3 Layout segmentation in Pre-deep learning era

In the era before the widespread adoption of deep learning, layout segmentation was a technique in computer vision and image processing. Its primary objective was to partition a document or image into distinct regions or segments based on the arrangement of visual elements, including text, images, graphics, and other visual components. This segmentation process played a vital role in enabling the interpretation of the spatial organization of these elements within a document or image. It also paved the way for various tasks such as optical character recognition (OCR), document categorization, and content extraction. Researchers typically categorise layout segmentation methods into three main approaches: Bottom-up, Top-down, and Hybrid.

2.1.3.1 Bottom-up approach

In the Bottom-up approach, the segmentation of a document or image commences with analysing lower-level visual features and components, such as individual pixels, lines, or regions. This process involves examining and grouping these foundational elements based on their visual characteristics, similarities, or other relevant criteria. As a result of this hierarchical grouping, higher-level segments or regions are gradually formed, ultimately leading to the complete segmentation of the document or image. The term "bottom-up" denotes the method's starting point at the most fundamental visual components, allowing for the potential capture of the document's hierarchical structure and layout. Some of the bottom-up approaches are as follows:

- **Docstrum:** The document spectrum, or docstrum, is a structural page layout analysis method [12]. The approach determines a page's format by analyzing the connected components. The docstrum method involves bottom-up, nearest-neighbour clustering of page components to accurately measure skew, within-line and between-line spacings and locate text lines and text blocks. It offers advantages over other methods, such as independence from skew angle and different text spacings and the ability to process local regions of different text orientations within the same image. The docstrum method is versatile and effective in analyzing various page formats and randomly oriented subpages.

  However, these approaches come with certain drawbacks:

  - **Computational Complexity** The docstrum method tends to be more computationally intensive, especially when dealing with a larger number of image components, making it comparatively slower than top-down methods.
  - **Skew Estimation Precision** In cases where documents exhibit only slight skew, the precision of skew estimation in the docstrum analysis may be inferior to that of baseline methods.
  - **Block Segmentation** The block segmentation stage of the docstrum analysis is less robust when compared to text line detection. In scenarios where the spacing between functional blocks becomes similar to the spacing within a block, segmentation requires more than

just structural layout analysis. Additional page-specific formatting information becomes necessary.

  - **Handling Joined Characters** The docstrum method may not accurately handle joined characters in handwriting or characters that lack clear separation.
  - **Lack of Skew Precision Comparison** The docstrum method may not accurately handle joined characters in handwriting or characters that lack clear separation. The docstrum analysis must provide a means to compare skew estimation precision against different methods, making it challenging to assess its accuracy relative to other techniques.
  - **Dependency on Connected Components** The docstrum method relies on connected components for analysis, which may not be suitable for all page layouts, potentially limiting its applicability in some instances.

- **Area Voronoi Diagram:** Kise *et al.* [13] introduce an approach to page segmentation utilizing an approximated area Voronoi diagram. This method facilitates the identification of potential document component boundaries in page images characterized by non-Manhattan layouts and skew. This approach leverages the identified candidates to estimate intercharacter and interline gaps without requiring domain-specific parameters. This design choice enhances the method's flexibility and efficiency. Experimental findings validate the method's efficacy in extracting body text regions, demonstrating efficiency levels comparable to other connected component analysis methods. The paper underscores the significance of page segmentation in the broader context of document image understanding, particularly in page classification. It addresses the growing prevalence of pages featuring non-Manhattan layouts and document components with arbitrary shapes. The drawback of this approach is as follows:

  - The proposed approach frequently results in the fragmentation of figures, tables, halftones, and titles featuring larger fonts, headers, and footers with wider interword gaps.
  - It is important to note that the method does not consider ruled lines that may be present in page images. This omission can result in segmentation errors, especially in low-resolution images, where it may break solid ruled lines or incorrectly filter out dotted ruled lines as noise.
  - While the method proves effective in extracting body text regions, it may encounter challenges when accurately segmenting other document components, such as auxiliary text regions and non-text regions. The rates of fragmentation and over-merging, which can subsequently pose issues in further stages of document image understanding, are not explicitly addressed with these components.
  - The paper acknowledges that there is room for improvement in the method and suggests that additional research is required to address these limitations.

2.1.3.2 Top-down approach

Top-down document image segmentation is used in computer vision and image processing to divide a document or image into meaningful regions or segments based on a predefined set of rules or criteria. Unlike bottom-up segmentation, which starts with low-level visual features and progressively builds up to form segments, top-down segmentation begins with a high-level understanding of the document's structure and content. Top-down image segmentation is beneficial for scenarios where the document layout and structure are well-defined or follow specific conventions. It relies on a priori knowledge of document formats and content patterns to guide the segmentation process. However, it may encounter challenges when dealing with documents that deviate from established norms or exhibit complex and irregular layouts. Some of the approaches that belong to this category are as follows:

- **Run Length Smearing Algorithm** The Run-Length Smoothing Algorithm [14], abbreviated as RLSA, is a robust technique employed in document analysis systems for top-down block segmentation. This algorithm is specifically designed for binary images and leverages the presence of white runs in both horizontal and vertical directions.

  The RLSA method comprises three primary stages: horizontal smoothing, vertical smoothing, and an additional round of horizontal smoothing. During the horizontal smoothing stage, it eliminates white runs shorter than a designated threshold smoothing value (SV). This same process is applied vertically. The outcomes from horizontal and vertical smoothing are combined using a logical AND operation to generate a new smoothing image. The RLSA method functions as a top-down procedure, successively applied to isolate significant blocks such as paragraphs, images, text lines, words, and character blocks. Choosing appropriate smoothing values is paramount for each stage of this process. The Horizontal Smoothing Value (HSV) and Vertical Smoothing Value (VSV) are contingent upon the number of pixels spanning the length of long words. At the same time, an HSV should encompass a few character widths. While various document analysis systems have adopted the RLSA method, they have traditionally relied on heuristic approaches to determine the smoothing values, lacking a systematic calculation method.

  The main drawbacks of this approach are the following:

  - RLSA is sensitive to document skew, which can affect the accuracy of the segmentation process.
  - The RLSA method depends on heuristic parameters determined subjectively, potentially resulting in inconsistencies and failures if the parameters assumptions have the wrong value.
  - The RLSA method requires training the system with documents with similar fonts or morphological characteristics, making it less robust and adaptable to different documents.
  - RLSA imposes a smoothing process on the document using heuristic parameters, which may only sometimes yield optimal results and can lead to suboptimal segmentation.

- **X-Y cut** Nagy *et al.* [15] proposed the X-Y tree-based document image segmentation and labelling. The X-Y tree data structure is employed to partition the document page into nested rectangles. In this structure, each node represents a rectangular block, and its children denote subdivisions of the parent block in either the horizontal or vertical direction. This representation effectively captures the document's layout structure.

  Syntactic analysis plays a pivotal role in recognizing and parsing the document's layout and structure, encompassing elements like headings, paragraphs, lists, and other structural components. It facilitates the segmentation and labelling of different document parts based on their syntactic attributes. Specialized software tools, formal grammar, and parsing algorithms achieve syntactic analysis, interpreting the syntactic structure within text and images.

  The application of syntactic analysis progressively enhances the refinement of image blocks. This refinement entails converting the two-dimensional challenge of page decomposition into a series of one-dimensional problems. The approach identifies logical entities of interest within the document page by examining the page image's vertical and horizontal projections.

  Researchers employ block grammar to assign labels to entities like abstracts, title blocks, byline blocks, reference entries, and figure captions at the subblock level. Different document formats may employ alternative grammar. During the parsing of a block, each subblock receives a unique label.

  Block grammar extension encompasses the entire page's segmentation and labelling. Each subblock within a given block is interpreted as a specific labelled entity, enabling the identification and labelling of significant information blocks on the page. The X-Y tree approach strives to preserve the original document's layout and typography faithfully, which improves readability compared to OCR results that might retain only partial information. This is especially crucial for graphics, equations, and tables, which may need more standardized computer representations. However, the approach has the following limitations:

  - Document images often contain noise caused by flawed fibers, imperfect printing, photocopying, and digitization, which can complicate automated analysis. This can affect the accuracy of the segmentation process.
  - The X-Y tree approach relies on the layout structure of the document, which may vary across different document formats and styles. Hence, developing a universal approach that works effectively for all documents is challenging.
  - Syntactic analysis, a vital component of the X-Y tree approach, can be computationally intensive and require sophisticated algorithms. The complexity of analyzing the structure and grammar of sentences or documents can impact the efficiency and scalability of the segmentation process.
  - Incorporating OCR and image-grabbing techniques into the X-Y tree approach can enhance it, but the accuracy of these techniques can affect the quality of the segmentation results.

Errors in OCR or image grabbing can lead to incorrect labelling and segmentation of the document image.

2.1.3.3 Hybrid approach

The hybrid approach merges bottom-up and top-down techniques, resulting in a more comprehensive examination of page layouts. The hybrid method adeptly manages intricate non-rectangular layouts, as seen in contemporary magazine pages, while preserving an understanding of the overall structure. The hybrid approach effectively addresses the constraints of exclusively top-down or bottom-up methodologies, which may require assistance in dealing with various formats and the frequently encountered non-rectangular regions within magazine pages. In summary, the hybrid approach in page layout analysis provides a more resilient and adaptable solution for accommodating diverse page layouts.

- **Tab-Stop based Layout Analysis** Smith [16] introduces a hybrid document layout analysis using Tab-Stop detection. The algorithm aims to identify a page's column layout by detecting tab-stops used during formatting. Tab-stops are considered as useful features for finding the column structure of a page and can handle complex non-rectangular layouts of modern magazine pages. The hybrid approach overcomes the limitations of purely bottom-up or top-down methods by combining them, allowing for a more comprehensive analysis of page layouts. The algorithm imposes structure and reading order on the detected regions based on the column layout.

  Tab-stops are fixed x-positions on a page that define the boundaries of regions such as margins, column edges, indentation, and table columns. Tab-stops play a crucial role in distinguishing tables from body text and bounding non-column elements like inset images and pull-out quotes. This layout analysis algorithm utilizes tab-stop detection to identify the column layout of a page. By tracking text lines from one tab-stop to another, the algorithm connects tab-stops and associates them as likely opposite sides of a text column. Tab-stops provide useful features for page layout analysis and table detection, making them valuable for analyzing complex layouts.

  The main disadvantages of this approach are as follows:

  - While the proposed hybrid page layout analysis algorithm focuses on tab-stop detection and column layout, it does not encompass logical layout analysis tasks, including identifying headers, footers, body text, numbered lists, and the segmentation of content into articles.
  - The algorithm may encounter challenges in dealing with non-rectangular regions and cross-column headings that smoothly blend into the columns below. These difficulties arise because the algorithm primarily depends on bottom-up and top-down methods.
  - The algorithm assumes the presence of isothetic region polygons, characterized by edges alternating horizontally and running parallel to the mean tab line. However, this assumption may not accurately represent complex layout structures.

– The algorithm's utilization of stroke width calculations for filtering text-connected components (CCs) may not be universally practical, especially when dealing with various font types. This approach could lead to both false positives and false negatives.

Table 2.1 summarizes the pre-deep learning approaches on document image segmentation and their observations.

2.2 Deep Learning Approaches

This section delves into applying deep learning techniques in document image analysis, encompassing feature design and layout segmentation. The advent of the deep learning era has brought about a significant shift from traditional 'feature engineering' to 'feature learning,' driven by the resurgence of neural networks. An early example of this shift is in LeNet-5 [94], which introduced an end-to-end trainable pattern recognition framework that could take raw pixel data as input, learn feature representations, and then classify handwritten digits. While LeNet-5 provided the foundation for subsequent architectures, recent developments have led to the utilization of much deeper networks, improved optimization algorithms, larger datasets, and the utilization of enhanced computing power through graphics processing units (GPUs).

In the forthcoming subsections, we will delve into the core principles of CNN architecture and explore recent advancements, providing insights into how researchers have skillfully designed these architectures for document feature extraction and layout segmentation.

2.2.1 Convolutional Neural Networks

A Convolutional Neural Network (CNN) is a specific type of feed-forward network denoted as g that takes an input $x \in \mathbb{R}^p$ and maps it to an output $y \in \mathbb{R}^d$. To be more precise [95], $g = f_L \circ \cdots \circ f_1$ represents a sequence of more straightforward functions f_l, referred to as layers. Let us denote each network function's outputs as $x_0, x_1, \ldots, x_{L-1}$, with x_0 representing the input image. At a specific layer l, we can express the output as $f\left(x_{l-1}, w_l\right)$, where w_l denotes the layer parameters encompassing weights and biases. If we have an image $I \in \mathbb{R}^{H \times W \times F}$, where H, W, F denotes the image's height, width, and number of channels (typically 3 for RGB images), the output at the l-th layer of the CNN takes the form of a spatial map $x_l \in \mathbb{R}^{H_l \times W_l \times D_l}$. Here, H_l and W_l correspond to the spatial resolution, while D_l represents the number of feature channels in layer l. Various functional layers within the network perform fixed or learned functions to transform inputs from one space to another.

A convolutional layer performs convolution by convolving an input map x with a set of K multi-dimensional filters f, resulting in an output y. The dimensions of these tensors are as follows: x is in $\mathbb{R}^{H \times W \times F}$, f is in $\mathbb{R}^{H' \times W' \times F \times K}$, and the output y is in $\mathbb{R}^{H'' \times W'' \times K}$.

$$y_j^n = f\left(\sum_{k=1}^{F} x_k^{n-1} * w_{kj}^n\right), j = \{1, \ldots, K\} \tag{2.10}$$

In the given context, 'n' represents the layer index, 'k' is the input channel index, and $w_{kj}^{n} \in \mathbb{R}^{h \times w}$ corresponds to the j^{th} set of filter weights. The '$*$' symbol denotes the convolutional operator, and $f(.)$ signifies an activation function, with y_j^n representing the j^{th} output feature map. Convolution is a linear and translation-invariant operator acting as a local image detector in this context. It involves using 'K' filters, where each filter functions as a sliding window over local regions of the input from the previous layer, denoted as x^{n-1}. Notably, the parameters are shared across different local regions, reducing the number of learnable parameters. This approach leverages the stationarity principle of image signals and maintains pixel dependencies within local regions. To capture complex transformations in the input space, non-linear activation functions are applied after each convolution operation. Standard activation functions include sigmoid and $tanh$, but they can suffer from vanishing gradient issues, particularly in the regions of extreme values, leading to slower training. Rectified Linear Units (ReLUs) [96] address this issue by having non-saturating gradients, although they can result in "dead" units that learn a high negative bias. To address this, researchers introduce leaky ReLU units, which progressively alleviate this issue during training. Maxout [97] activation units are a more generalized version of ReLUs, approximating non-linear activation functions with multiple piece-wise linear units. Popular activation functions used in convolutional neural networks (CNNs) include:

1. Rectified Linear Unit (ReLU): ReLU is one of the most widely used activation functions. It replaces all negative values with zero and leaves positive values unchanged, which helps in mitigating the vanishing gradient problem.

2. Leaky ReLU: Leaky ReLU is a variant of ReLU that allows a slight, non-zero gradient for negative values. This helps prevent some neurons from becoming completely inactive during training.

3. Parametric ReLU (PReLU) [98]: PReLU is similar to Leaky ReLU but allows the negative slope to be learned during training, making it more flexible.

4. Exponential Linear Unit (ELU) [99]: ELU is another activation function that addresses the vanishing gradient problem. It smoothly saturates for negative values and has exponential growth for positive values.

5. Sigmoid: Sigmoid functions squash the input values into a range between 0 and 1. Researchers often use it in the output layer of binary classification models.

6. Hyperbolic Tangent ($tanh$): Tanh functions map input values to a range between -1 and 1. They are similar to sigmoid functions but have a zero-centered output.

7. Swish [100]: Swish is a relatively new activation function that combines elements of ReLU and sigmoid functions. It has shown promise in some deep-learning applications.

8. Maxout: Maxout is a generalization of ReLU that learns piece-wise linear activation functions and can approximate more complex functions.

The choice of activation function depends on the dataset's specific problem, architecture, and empirical performance. Experimentation is often necessary to determine which activation function works best for a given task.

The pooling layer is a fundamental component in a Convolutional Neural Network (CNN) used for feature extraction and dimensionality reduction. Its primary purpose is to reduce the feature maps' spatial dimensions (width and height) while retaining the most essential information. This spatial down-sampling helps reduce the network's computational complexity and makes it more computationally efficient.

One of the most common pooling operations is Max Pooling, where for each local region in a feature map (typically a small square window, like 2×2 or 3×3), the maximum value is retained while the rest are discarded. This process effectively reduces the spatial dimensions by selecting the most significant features. Another pooling operation is Average Pooling, which takes the average value within each local region, providing a smoothed down-sampled representation. Pooling layers are typically applied after one or more convolutional layers in a CNN, helping to create translation-invariant features, meaning that the network can recognize patterns regardless of their exact position in the input.

The pooling layer is vital in handling spatial hierarchies of features in CNNs. By iteratively applying pooling layers, the network can capture features at different scales and progressively reduce the spatial dimensions. This leads to a more compact representation that retains the critical information for classification or detection tasks. However, it is essential to strike a balance between pooling aggressively (which may result in loss of fine details) and not pooling enough (which may lead to excessive computational requirements). Fully connected layers follow pooling layers, allowing the network to leverage the high-level abstractions it has learned for tasks such as image classification, object detection, or semantic segmentation.

Fully connected layers, also known as dense layers, are crucial in many neural network architectures, including deep learning models like Convolutional Neural Networks (CNNs) and feedforward neural networks (FNNs). These layers are responsible for learning complex relationships and patterns within the input data, making them suitable for tasks such as image classification, natural language processing, and more.

In a fully connected layer, each neuron (or node) connects to every neuron in the preceding layer. If the N neurons are in the previous layer, N connections will go into each neuron in the fully connected layer. Each connection has an associated weight, adjusted during training to learn the best representations for the given task.

The primary purpose of fully connected layers is to transform the features learned in the previous layers into a form that's suitable for the final prediction or decision-making. Neural network architects often position these layers at the end of the network structure, and they subsequently apply an activation function (such as ReLU or sigmoid) to introduce non-linearity.

2.2.1.1 Neural Network training

In CNN network training, the goal is to determine the best weights for every trainable parameter within the network. These parameters consist of filter weights (referred to as W) associated with convolutional layers and biases (referred to as b) associated with both convolutional and fully connected layers. To achieve this, researchers frame the learning process as an optimization problem centered around minimizing the following objective function:

$$\theta^* = \underset{\theta}{\operatorname{argmin}} \sum_{n=1}^{N} \mathcal{L}\left(x^n, y^n\right) + \lambda \|W\|_2^2 \tag{2.11}$$

In this context, the objective function involves a loss function denoted as $\mathcal{L}(\cdot)$, which depends on a parameter set represented by θ. This parameter set encompasses both W and b, where W signifies the filter weights, and b represents the biases. The variables x^n and y^n refer to the input sample and the corresponding ground truth label vector for the nth sample. The latter part of the equation focuses on minimizing the $L2$ norm of the weight coefficients, commonly referred to as weight decay. This regularization technique aims to counteract overfitting by reducing the complexity of the network, essentially acting as a form of regularization.

Typically, the loss function for a deep convolutional network featuring non-linear activation functions is notably intricate and lacks convexity. The prevailing optimization approach of choice is the mini-batch stochastic gradient descent (SGD) with momentum. SGD is a first-order iterative optimization technique that calculates the gradient of the loss function and then iteratively adjusts the weights in a direction that reduces the loss. This adjustment process can be described as follows:

$$\theta_{n+1} = \theta_n - \eta\Delta \tag{2.12}$$

$$\Delta = \mu\Delta + \frac{\partial L}{\partial \theta_n} \tag{2.13}$$

In this context, η represents the learning rate, determining the magnitude of each update. In contrast, μ represents the momentum factor, influencing the incorporation of the gradient direction from the previous step into the current update. It is worth noting that several weight initialization techniques [98, 101] exist to facilitate the learning of optimal weights.

To update the parameters of individual layers within the CNN network, the backpropagation algorithm [102] is employed, leveraging the principles of the chain rule for derivatives. Let us examine the intermediate computational block [95] within the CNN network, denoted as layer f with parameters w, taking input x and producing output y. To facilitate clarity in our explanation, we have omitted the inclusion of an activation functional block. In this context, the function denoted as h calculates the loss, resulting in a scalar output represented by z. The derivative of the composition $h \circ f$ concerning both x and w is expressed as follows:

$$\frac{\partial z}{\partial w} = \frac{\partial z}{\partial y}\frac{\partial y}{\partial w} \tag{2.14}$$

$$\frac{\partial z}{\partial x} = \frac{\partial z}{\partial y}\frac{\partial y}{\partial x} \tag{2.15}$$

The derivatives mentioned above are propagated backward through each intermediate layer to the input layer, leveraging the chain rule. These principles apply to all directed acyclic graphs (DAG) with differentiable layer functions. To summarize the training process, we rely on two computational pathways:

1. **Forward Pass**: This involves taking input x and parameters w to compute the output y.
2. **Backward Pass**: In this step, we utilize input x, parameters w, and $\frac{\partial z}{\partial y}$ to calculate $\frac{\partial z}{\partial w}$ and $\frac{\partial z}{\partial x}$.

After computing these gradients, we can apply the Stochastic Gradient Descent (SGD) algorithm, as shown in Equation 2.13, to update the parameters.

2.2.1.2 Loss Calculation

The loss function assesses the alignment between predictions and the actual ground truth. Researchers and practitioners commonly employ several well-known loss functions for tasks such as classification and retrieval:

- Cross-entropy loss: Given by $L = -\sum_i y_i \log(\hat{y}_i)$, this function employs one-hot encoding (y_i) for the i^{th} class and $\hat{y}_i$ represents the predicted probability of the i^{th} class after the softmax layer.
- Mean square loss: Expressed as $L = \frac{1}{2}\|y_i - \hat{y}_i\|_2^2$, this loss measures the squared difference between the ground truth (y_i) and predicted ($\hat{y}_i$) values.
- Hinge loss: Defined as $L = \max(0, 1 - y_i\hat{y}_i)$, this loss captures the maximum of zero and the difference between the actual class label (y_i) and predicted score ($\hat{y}_i$). Notably, hinge loss may not always be differentiable, but sub-gradients exist.

In addition, various other loss functions like negative log-likelihood, KL divergence, cosine embedding, margin-based ranking loss, etc., cater to different problem domains. Most of these loss functions are continuous and differentiable, with hinge loss being an exception due to its sub-gradient nature.

2.2.1.3 Popular Architectures

Popular convolutional neural network (CNN) architectures represent a family of deep learning models widely adopted and achieved state-of-the-art results in various computer vision tasks. Here, briefly explain some of the most popular CNN architectures:

1. **LeNet-5:** LeNet-5 [94], developed by Yann LeCun in the early 1990s, was one of the earliest CNN architectures. It was primarily designed for handwritten digit recognition and was foundational in developing CNNs. LeNet-5 introduced the concept of convolutional and pooling layers.

2. **AlexNet:** Developed by Alex Krizhevsky and his team [103], AlexNet won the ImageNet Large Scale Visual Recognition Challenge in 2012, significantly advancing the field. It consists of multiple convolutional layers, max-pooling and fully connected layers. AlexNet's success popularized deep learning and the usage of CNNs for image classification.

3. **VGGNet:** The VGG architecture [104], developed by the Visual Geometry Group (VGG) at the University of Oxford, is known for its simplicity and uniform architecture. It consists of 16 or 19 weight layers with small 3x3 convolutional filters make it easy to understand and train. VGGNet achieved excellent results on image classification tasks.

4. **GoogLeNet (Inception):** GoogLeNet [105] introduced the concept of inception modules, which use multiple convolutional filters of different sizes within a single layer. This architecture reduces the number of parameters while increasing network depth. GoogLeNet won the 2014 ImageNet competition with impressive accuracy.

5. **ResNet:** Residual Networks, developed by Kaiming He and his team, introduced residual connections [106]. These connections allow deep networks (hundreds of layers) by mitigating the vanishing gradient problem. ResNet architectures, such as ResNet-50 and ResNet-101, have become the basis for many state-of-the-art models.

6. **DenseNet:** Dense Convolutional Networks (DenseNet) [107] connects each layer to every other layer in a feedforward fashion. This architecture encourages feature reuse and alleviates the vanishing gradient issue. DenseNet has shown excellent performance while being more parameter-efficient.

7. **MobileNet:** MobileNet [108] efficiently deploys mobile and embedded devices. It employs depth-wise separable convolutions to reduce the computational cost while maintaining accuracy. MobileNet variants, like MobileNetV2 and MobileNetV3, provide different trade-offs between speed and accuracy.

8. **EfficientNet:** EfficientNet [109], proposed by Tan and Le in 2019, uses a compound scaling method to balance network depth, width, and resolution, resulting in highly efficient models in terms of both computational resources and accuracy. It has set the standard for efficient CNN architectures.

These CNN architectures have significantly advanced the field of computer vision and deep learning, making them a critical part of modern AI systems for tasks like image classification, object detection, and segmentation. Researchers and practitioners often choose these architectures based on the specific requirements of their applications, balancing model complexity, accuracy, and computational efficiency.

2.2.2 Deep features

Deep features represent data derived from deep neural networks, particularly Convolutional Neural Networks (CNNs) and other deep architectures. These representations capture hierarchical and abstract information from the input data, which proves valuable across a range of machine learning tasks, notably in computer vision, document image analysis, and natural language processing. CNNs consist of multiple layers, including convolutional, pooling, and fully connected layers, each extracting increasingly abstract and complex features. Early layers identify basic features like edges and textures, while deeper layers recognize more intricate patterns such as fonts, scripts, and document layouts.

Transfer learning is a typical application of deep features. Pre-trained deep neural networks, like CNNs trained on datasets such as ImageNet [110] or PubLaynet [41], have learned meaningful features. These features can be adapted and fine-tuned for other tasks, even when limited task-specific data is available. This approach is potent for achieving strong performance in tasks like image classification, object detection, or document layout analysis without building models from scratch. Deep features often have lower dimensionality than raw data, effectively reducing the complexity of machine learning algorithms. In various applications, deep features are extracted from the hidden layers of pre-trained neural networks and serve as input for traditional machine learning models such as support vector machines (SVMs) [111] or random forests [112], creating a synergy between deep learning and pre-deep learning approaches.

These features capture the high-level semantics of data. In natural language processing, they represent the meaning of words and phrases through word embeddings. In computer vision, deep features can signify object classes, poses, or even emotions in facial recognition. In document image analysis, they describe font types, scripts, figures, and document layouts. Moreover, techniques like t-SNE (t-distributed stochastic neighbour embedding) [113] allows for the visualization of deep features, providing insights into what the network has learned and enabling clustering and pattern recognition.

In summary, deep features are a cornerstone in solving complex document image analysis problems, revolutionizing fields such as computer vision and natural language processing. They facilitate the automatic extraction of rich, hierarchical representations from raw data, substantially enhancing the performance of diverse applications.

2.2.2.1 Mix and Match deep features

Researchers [114, 115] have explored combining pre-deep learning methods with deep learning approaches to develop efficient feature representations. They introduced a novel "mix and match" technique for deep feature extraction.

Girshick *et.ãl* [115] introduced a novel approach named R-CNN (Regions with CNN features), which aims to create a comprehensive hierarchy of image features. This is achieved by integrating region proposals with convolutional neural networks (CNNs), enhancing object detection capabilities. Combining high-capacity CNNs with these bottom-up region proposals enables the precise localization and segmentation of objects, ultimately improving object detection performance. Their work also highlighted

the effectiveness of supervised pre-training for an auxiliary task, followed by fine-tuning specific to the target domain. This strategy proved highly beneficial, particularly in scenarios with limited labelled training data, and contributed to significant performance enhancements. Furthermore, their study unveiled the existence of a complex hierarchy of image features that the network learns during the training process, shedding light on the insights gained from the network's internal representations. They proposed the existence of hierarchical, multi-stage processes involved in computing more informative features for visual recognition, offering a valuable perspective on the advancements in this field.

Cimpoi *et. al* [114, 115] investigated the recognition of material and describable texture attributes in complex environments. They proposed a novel texture descriptor, FV-CNN, derived by applying Fisher Vector pooling to a filter bank within a Convolutional Neural Network (CNN). FV-CNN significantly enhances texture, material, and scene recognition performance, surpassing the existing state-of-the-art methods. An advantage of FV-CNN is its ability to transfer seamlessly across domains, eliminating the need for feature adaptation as required by methods relying on the fully connected layers of CNNs. Additionally, FV-CNN can effectively incorporate multiscale information and describe regions of various shapes and sizes.

While R-CNN features focus on capturing the structure, shape, or arrangement of documents, FV-CNN features are adept at capturing the inherent texture characteristics within an image. These two sets of features possess a complementary nature, suggesting that their combination has the potential to enhance accuracy across a range of problems and applications.

2.3 Features used in document image analysis

2.3.1 Image Classification based on Texture Feature

In recent years, texture feature analysis has undergone tremendous growth. It plays an essential role in the study of various kinds of images. For several decades, handcrafted texture features have been popularly used to analyse several document image types in the literature. The underlying assumption for the methods based on texture features is that the textual regions in any digitized documents are considered textured areas. In contrast, their non-text contents are considered regions with distinct textures [116]. Furthermore, the text with different fonts is also distinguishable due to its textual features. Numerous hand crafted texture features — Tamura [117], Local Binary Pattern (LBP) [118, 119], Gray-Level Run-Length Matrix (GLRLM) [120], Auto-Correlation [121], Gray-Level Co-occurrence Matrix (GLCM) [122], Gabor filters [123] and Wavelet transformation [124] are successfully applied on analysis of various kinds of document images. Examples like — font recognition [125], handwritten classification [126, 127], printed script identification [128–133], segmentation of historical documents [134–137], segmentation of printed documents [138–140], document image classification [141–143].

Texture features extracted from GLCM are used to solve script identification problem in different levels - page [131, 144, 145], text line and word [146] on multi-script document images. Furthermore, wavelet coefficient based texture descriptor is also employed to identify script in different levels like

page [147–150], text line [132, 151], word [152], and character [153, 154] of multi-lingual document images. The researchers used Gabor filter based texture descriptor to identify script in page [128, 130, 131, 144, 145, 155–158], word [133, 159–161], and character [162–164] levels on various multi-lingual documents. Rotation invariant texture feature extracted from multi-channel Gabor filter is applied to identify script in a multi-lingual printed documents [130].

The performance of a texture feature designed for a specific task degrades for other tasks. It is because of the lack of generalization capabilities of handcrafted texture features. In contrast, the features learned from a large set of training images using deep learning have better generalization capability [165]. These admirable characteristics of deep features have led to the invention of several deep learning-based algorithms to solve different tasks on document image analysis [45, 166–176]. These works experimentally establish that deep feature is superior to handcrafted features for solving various tasks on document images.

In classical computer vision, textures can be recognized using filter bank responses [177–179], scale invariant feature transform (SIFT) [86, 180], Bag-of-Words (BoWs) [181] using dictionary learning, and the multi-texton histogram (MTH) integrates the advantages of co-occurrence matrix and histogram by representing the attribute of co-occurrence matrix using histogram [182]. Cimpoi *et al.* [183] proposed Fisher Vector encoded convolutional neural network to extract texture features and achieved state-of-the-art performance. In this work, the authors analyzed the performance of two individual features (i) texture feature (FV-CNN), (ii) global feature (FC-CNN) and their combination for the natural image segmentation task. The main drawback of this work is that the network is not end-to-end trainable. Jobin and Jawahar [40] successfully applied this approach to segment historical document images. In this direction, Zhang *et al.* [184] proposed an end-to-end trainable texture encoding network with a novel encoding layer integrated on top of convolutional layers. Dictionary learning and encoding have been done in a single module. This network has no global feature extraction head, which is necessary for a general-purpose network.

2.3.2 Image Classification based on Discriminative Patch

Discriminative patches are the key patches that help make the classification decision accurately based on subtle differences. Earlier works from this category [185, 186] depend on additional semantic part annotations, while more recent ones [187–189] only require category labels. These [187–189] motivate us to design an image classification network that learns the most discriminative part of an image and encodes the texture nature separately.

2.3.3 Image Classification based on Global Feature

Global features or shape-preserving features are commonly used for classifying document images. Several studies on document image classification using hand-crafted global features have been done in the recent past [190,191]. A related survey [141] discussed the performance of hand-crafted features and classifiers for document image classification tasks. Large variations in content, style and layout make it

difficult to achieve good accuracy using these approaches for generic document classification. Recently, numerous approaches [45, 166–168, 192] have been developed based on deep features to improve the accuracy in the classification of generic document images.

Kang *et al.* [192] considered a convolution neural network (CNN) to automatically learn the hierarchical layout features to classify document images into predefined categories. In the same direction, Harley *et al.* [166] explored the hierarchical feature learning capability of CNN for document image classification tasks. Here, CNN is trained on non-document images and transferred to document analysis tasks. Furthermore, Kolsch *et al.* [168] proposed a deep network to extract features and then considered extreme learning machines for classifying document images. Replacement of fully connected layers by extreme learning machine makes classification faster and suitable for real-time applications. Afzal *et al.* presented an exhaustive investigation of deep learning architectures, algorithms, and strategies for document image classification in [45]. The authors used deep neural network architectures including — VGG-M [193], VGG-V [104], ResNet [106] and GoogleNet [194] and transfer learning (from real images) for document image classification. Similarly, Tensmeyer and Martinez [167] analyzed the deep network architectures, data augmentation, and training strategies to classify document images. These deep learning-based approaches performed better than hand-crafted texture feature-based techniques for the document image classification tasks. These deep architectures extract global image features that are sometimes ineffective for document classification due to the large inter-class document similarity. The deep network we designed extracts texture and local discriminative features, supplementary to the global feature for fine-grained document image classification. Table 2.1 depicted the various feature extraction approach and its observations.

2.3.4 Document Image Segmentation

Various methods have been presented in the literature to perform the document image segmentation task. Most of these approaches focus on improving any of the stages of the segmentation pipeline, such as pre-processing, feature extraction, feature modeling, etc.

The work [195], presents an entropy-based segmentation algorithm for document images. The algorithm eliminates the noise inherent to the document itself, specially in documents written on both sides. The algorithm presented in [196] uses two background light intensity normalization algorithms to enhance the quality of images before a local adaptive binarization algorithm is applied. The image normalization algorithms use adaptive linear and non-linear functions to approximate the uneven background of the images due to the uneven surface of the document article, aged color and light source of the cameras for image lifting.

The recent article on historical document images [197] utilizes the idea of an autoencoder to learn the features automatically. Generally, an autoencoder is a neural network trained to reconstruct its input. In this work, the encoder output of the autoencoder is extracted as features and fed to an off-the-shelf classifier to segment an image. Training an autoencoder is a time-consuming task, and in [198], the SLIC superpixel extraction stage was incorporated to speed up the process in [197]. Another work [199]

uses a Conditional Random Field (CRF) [200] to model the local and contextual information jointly. It helps to refine the segmentation results of [198]. Another recent approach [178] segments document images with a simple CNN architecture. Inspired by [198], we consider the segmentation problem as a pixel labeling problem. In contrast to the above-discussed approaches, our work extracts deep features from each superpixel and classifies them using SVM. Our results show that deep features are more robust than handcrafted features, the autoencoder-based approach [197] and the CNN-based approach [178].

Textual regions and graphics regions dominate document images. Graphics include natural images, various graphs, plots, and tables. Historically, the document image segmentation task is considered dividing a document into various regions. Existing techniques [201, 202] are either bottom-up or top-down. In the bottom-up [201] case, individual words are first detected based on hand-crafted features. Therefore, detected words are grouped to form lines and paragraphs. Segmenting textual regions (like paragraphs) into lines and words helps to find semantic regions present in the document in the case of top-down [202]. These methods mostly use feature engineering and do not require training data while applying clustering or thresholding.

Several neural-based models have been proposed to segment document images with the recent advancement in deep convolutional neural networks (CNNs) [170, 197, 203–205] have been proposed to segment document images. Chen *et al.* [197] considered a convolutional auto-encoder to learn features from cropped document image patches. Thereafter, those features are used to train an SVM classifier [111] to segment document patches. Vo *et al.* [203] and Renton *et al.* [204] proposed a lines detection algorithm in the handwritten document using FCN. In [205], Yang *et al.* proposed an end-to-end, multi-modal, fully convolutional network for extracting semantic structures from document images. Inspired by document image segmentation and natural image segmentation using deep CNNs, Haurilet *et al.* [7, 8] explore DeepLab [29] architecture on classroom slide segmentation tasks. More recently, DANet [10], HANet [9], and DRANet [11] capture long-range dependency to improve performance by extending the self-attention mechanism. The proposed LEANet is strongly influenced by DANet [10], HANet [9] and DRANet [11].

Textual regions and graphics regions dominate document images. Graphics include natural images, artificial images like graphs, plots and tables. Historically, the document image segmentation task is considered as segmentation of the document into various regions. Existing techniques [206–208] are either bottom-up or top-down in nature. In the case of bottom-up [206], individual words are first detected based on handcrafted features. Therefore, detected words are grouped to form lines and paragraphs. Segmenting textual regions (like paragraphs) into lines and words helps to find semantic regions in the document in the case of top-down [207, 208].

Analysis of logical structure presented in document images is another way of segmentation. Mao *et al.* [209] defined logical structure in documents as a hierarchy of logical components, such as titles, headings, paragraphs and lists. Early works [210] in the analysis of coherent structure in document images have been done using a set of heuristic rules based on text location, font, etc. Instead of rule-based

approaches, learning-based techniques [211] have been developed to understand the logical structure of the documents. Strategies based on learning produce better results than rule-based methods.

Features play an important role in document image segmentation tasks. Various hand-crafted features [212, 213] were successfully considered for segmenting document images. With the recent advancement in deep convolutional neural networks, several neural-based models [197, 205, 214] have been proposed to segment document images. Chen *et al.* [197] considered a convolutional auto-encoder to learn features from cropped document image patches. In [205], Yang *et al.* proposed an end-to-end, multi-modal, fully convolutional network for extracting semantic structures from document images. Li *et al.* [214] proposed an effective method for page segmentation using convolutional neural network (CNN) and graphical model. The CNN extracts visual features, while the graphical model explores the relationship between visual features and regions.

2.3.5 Document Image Classification

The researchers solve the document image classification tasks using various types of features — (i) visual feature, (ii) textual feature, (iii) combination of textual and spatial features, and (iv) combination of visual, textual, and spatial features. The approaches using only visual features used no external feature extraction modules such as OCR and layout detector. Harley *et al.* [166] solve document image classification tasks using deep Convolution Neural Networks (CNNs). The researchers improve the performance of the document image classification task using various CNN architectures such as VGG-16 [45], AlexNet [167], GoogLeNet [215], InceptionResNetV2 [216], and LadderNet [217]. There is a limitation to improving accuracy by changing network architectures. Das *el at.* [44] use a stacked generalized ensemble technique to combine predictions generated by different base networks.

Other than visual features, several methods explore textual and spatial features for classifying document images. Xu *et al.* [218] successfully utilize the language representation model (BERT) [219] for classifying document images. This model interacts with the textual and layout features to improve classification accuracy. Works [218, 220, 221] combine visual, textual, and spatial features for document image classification and obtain the best results. All the above-discussed methods utilize only global visual features and not local discriminative and texture features for document image classification. Instead of global visual features, sometimes local discriminative and texture features help to discriminate one category of documents from other categories. Our work aims to extract three visual features and judicially combine them for the classification task.

2.3.6 Genre Classification from Book Cover

In recent years, there has been an increasing interest in automated genre classification based on images by leveraging the strength of the deep neural network. Iwana *et al.* [42] introduce a Book-Cover dataset with 30 genres and solve it using the AlexNet [165] pre-trained on ImageNet [165]. Zujovic *et al.* [222] classify paintings based on genres. The authors utilize gray-level features and color features from the images and feed these features to different classifiers for prediction. Holly *et al.* [223] use

textual features and visual features in a transfer learning framework to classify the genre of the book covers. In the same direction, Biradar *et al.* [224] use both the textual and image features to determine the genre of the book. The work [225] benchmarks a set of state-of-the-art image classification models for book cover classification.

2.3.7 Document Figure Classification

Figure classification is the primary step for information retrieval/extraction from a figure in a document image. Various figures like charts, tables, and natural images are used to visually represent a wide range of textual information in books, scientific articles, newspapers, etc. Text recognition using Optical Character Recognition (OCR) is the primary process for understanding the content of the document images. Increasing use of figures in documents suggests figure classification can be an important sub-task for OCR for a better and complete understanding of the document images [226]. In early works [31, 35, 37, 53, 54], different handcrafted features are used to recognize various types of charts in the document images.

Zhou *et al.* [53,54] considered Hough transformation to recognize bar charts in the document images. Prasad *et al.* [35] considered SIFT and HOG features to recognize five different types of chart images. Due to the large visual similarity among subordinate categories, the handcrafted features fail to achieve good accuracy on the figure classification task.

To solve the limitation of handcrafted features for the figure classification task, Kavasidis et al. [227] recently proposed a saliency-based Convolutional Neural Network (CNN) for localizing different types of figures in the documents. This work is limited to localize tables, Bar charts, and Pie charts. Tang *et al.* [33] proposed a novel framework (DeepChart) to classify charts by combining (CNNs) and Deep Belief Networks (DBNs). The authors experimentally established that their method is far better than handcrafted features. In the same direction, Siegel *et al.* [34] proposed various document figure classification algorithms using learnable features. Similarly, the work [228] used a deep neural network to rank the figures. The work [229] focuses on the full-text indexing of all text referring to the images and filtering for disciplines and image types.

2.3.8 Script Identification

Script identification is an inevitable step for text understanding under multi-lingual scenarios. The main challenge for script identification is the interference caused by the similarity between texts in different languages. Other are complex background, noise, and low resolution. Pre-deep learning approaches used handcrafted features to identify the script. Shijian *et al.* [230] proposed a document vectorization technique that transformed documents into electronic document vectors. Therefore, scripts/languages are identified using vertical component and shape analysis. The properties of the connected component are used to identify the script in [231]. Similarly, Sharma *et al.* [232] extracted local binary pattern, histograms of oriented gradients, and gradient local auto-correlation features and exploited SVMs and ANNs to classify the scripts.

Advancement deep neural architectures motivate researchers to use networks for script identification. Mei *et al.* [233] integrated CNN and LSTM-RNN to identify the scripts of natural scene images. Lu *et al.* [234] discussed a method for script identification by integrating the local and the global CNN features. The local features help to extract subtle differences from the scripts. The final decision was obtained by combining the results using the AdaBoost algorithm. Shi *et al.* [52] introduced a discriminative convolution network that can identify subtle discriminative features for the script. Bhunia *et al.* [235] used an attention-based Convolutional-LSTM network for script identification. Ghosh *et al.* [236] proposed a lightweight script identification network to identify video scripts.

2.3.9 Semantic Segmentation

Pixel-wise classification tasks like semantic segmentation achieve significant improvement inspired by replacing the fully connected layer in the image classification network with the convolution layer called a fully convolution network (FCN) [26]. Limitations of FCN model [26] are ignoring small objects and mislabeling the large objects due to fixed receptive field size. To address this issue, Noh *et al.* [237] proposed a semantic segmentation algorithm by learning a deconvolution network composed of both deconvolution and unpooling layers. Several methods [29, 238] used dilated convolution to enlarge the receptive field of the neural network. Since in deep networks, higher-layer feature contains more semantic meaning and less location information. Various methods [26, 238] have been developed by combining multi-scale features to improve the segmentation performance. The reader can find various semantic segmentation models in [239–241]. Table 2.1 summarizes the deep learning approaches on document image segmentation and their observations.

Table 2.1: Overview of the feature extraction and layout segmentation works with its approach and observations

Approach	**Key Idea**	**Observations**
Feature Extraction		
Bag of Words (BoW) / Bag of Visual Words (BoVW) [82, 83]	BoW/BoVW is a model for visual data analysis that represents an image as a collection of visual words analogous to a text document being a collection of words. It involves detecting keypoints, computing descriptors, creating a visual vocabulary and representing images as histograms of visual words.	This model assumes independence between visual words and does not account for spatial relationships between them, which can limit its ability to capture the structure of the visual data.

Approach	Key Idea	Observations
Interest Point Detection	In the BoVW framework, interest points or keypoints are extracted to compute descriptors. Methods include the unsupervised discovery of mid-level discriminative patches [84] and identifying distinctive components for scene classification [85]. Repeatability is a crucial characteristic for robustness against noise and degradation.	Selecting the optimal detector for a specific domain the task can be challenging. Some detectors may not be robust under various conditions, and dense sampling can be computationally expensive.
Keypoint Detectors: Harris Affine Detector [60]	A keypoint detector that specializes in detecting corner points and preserving those with higher response values. It is commonly used in feature detection for image processing.	Sensitive to changes in scale and rotation, which can affect its robustness in varying conditions.
Keypoint Detectors: Difference of Gaussians (DoG) [86]	Part of the SIFT descriptor, the DoG the detector identifies local maxima within a scale-space created by the difference of Gaussian pyramids, useful for identifying keypoints at multiple scales.	It may produce a large number of keypoints, including redundant ones, leading to increased computational complexity.
Dense Keypoints [87, 88]	Uses dense or regularly distributed keypoints across various scales, providing robust performance in various applications by not relying on specific feature points.	It can be computationally intensive due to the large number of keypoints processed, especially in high-resolution images.
Hierarchical Clustering [89]	A clustering method that builds a hierarchy of clusters, which can be represented in a tree-like structure. It helps reduce computational complexity in large datasets.	Can be computationally expensive for very large datasets due to the need to calculate all pairwise distances.
Approximate k-Means using Randomized kd-Trees [90]	An approximate clustering method using randomized kd-trees for efficient nearest neighbor search, significantly reducing the complexity of assigning a sample to a cluster.	While efficient, it may not achieve the same accuracy as exact k-means clustering, especially when clusters are not well-separated.
Vector Quantization (VQ)	A coding technique where each descriptor is assigned to its nearest codeword from the codebook, using a hard quantization approach.	It leads to a loss of geometrical information, reducing the ability to build robust image retrieval or recognition systems.
Spatial Pooling [91]	A pooling method that divides an image into spatial pyramids and combines weighted histograms from individual regions, aiming to regain some lost geometrical information.	Only partially regains geometric information, and the effectiveness depends on the chosen pyramid levels and weightings.

Approach	Key Idea	Observations
Vector of Locally Aggregated Descriptors (VLAD) [92]	An encoding method that aggregates the differences between local descriptors and their assigned visual words, capturing the first-order moment.	Higher dimensionality than traditional coding schemes, increasing computational requirements.
Fisher Vectors (FV) [93]	An encoding method applying Fisher Kernels to visual vocabularies, capturing the first, second, and third moments, providing a dense image description.	Computationally expensive due to the use of Gaussian Mixture Models (GMMs) and the calculation of Fisher information matrices.
Document Layout segmentation		
Bottom-up approach		
Docstrum [12]	The document spectrum, or docstrum, is a structural page layout analysis method. It involves bottom-up, nearest-neighbour clustering of page components to measure skew, within-line and between-line spacings, and locate text lines and text blocks. It can handle different text orientations within the same image.	- Computationally intensive, especially with more image components. - Less robust block segmentation, especially with similar spacing between blocks. - May not handle joined characters accurately in handwriting. - Depends on connected components, limiting applicability for all layouts.
Area Voronoi Diagram [13]	Introduces a page segmentation method using an approximated area Voronoi diagram to identify document component boundaries in non-Manhattan layouts and skewed pages. Efficient in extracting body text regions.	- Fragments figures, tables, titles with larger fonts, and wider interword gaps. - Does not account for ruled lines, potentially causing segmentation errors in low-resolution images. - May struggle with accurate segmentation of the auxiliary text and non-text regions.

Approach	Key Idea	Observations
Top-down approach		
Run Length Smearing Algorithm (RLSA) [14]	A top-down block segmentation method for binary images, smoothing white runs horizontally and vertical directions to isolate blocks like paragraphs, images, and text lines. Requires choosing appropriate smoothing values.	- Sensitive to document skew, affecting segmentation accuracy. - Relies on heuristic parameters, potentially causing inconsistencies. - Requires training with similar document fonts or characteristics, limiting adaptability.
X-Y Cut [15]	Uses a tree-based data structure to partition document pages into nested rectangles, aiding in segmentation and labelling based on document layout structure. Applies syntactic analysis to refine image blocks and recognize document layout.	- Noise in document images can complicate automated analysis. - Dependent on document layout, which may vary, limiting a universal approach. - Syntactic analysis can be computationally intensive and complex.
Hybrid approach		
Tab-Stop based Layout Analysis [16]	A hybrid approach using tab-stop detection to identify column layout in documents. Combines bottom-up and top-down methods to manage complex layouts and impose structure based on column layout.	- Does not include logical layout analysis like identifying headers or footers. - May struggle with non-rectangular regions and cross-column headings. - Assumes isothetic region polygons, which may not represent complex structures accurately.
Object detection approaches		
Layoutparser [242]	It offers a library of models, including Faster R-CNN and Mask R-CNN, trained on various document datasets such as HJDataset [243], PubLayNet [41], PrimaLayout [43], NewspaperNavigator [244], and TableBank [245]. The library is easy to integrate into document analysis tasks.	All the models are designed for object detection, so preprocessing steps like non-maximal suppression of the output are necessary. In contrast, transformer-based approaches offer greater flexibility and more accurate results compared to traditional detection-based methods.
Transformer-based approaches		

Approach	Key Idea	Observations
Shehzadi *et. al* [246]	The decoder utilizes a deformable attention mechanism to improve the efficiency of both self-attention and cross-attention processes. Additionally, it employs contrastive denoising on the object queries, which helps the model learn more quickly.	Because this approach combines both CNN and transformer networks, it is computationally intensive and involves complex learning procedures.
DLAFormer [247] is an end-to-end transformer model designed for comprehensive document layout analysis.	It unifies multiple layout analysis tasks, including text region detection, text line extraction, and relation prediction, within a single framework. By consolidating these tasks into a unified model, DLAFormer enhances efficiency and consistency in processing complex document structures, eliminating the need for separate models for each task.	The model exhibits limited generalization capabilities, particularly when applied to diverse or unseen data, which hampers its effectiveness in real-world applications
Language-based approaches		
LayoutLM [218] integrates both textual content and visual information, such as the coordinates of text blocks, to effectively understand document structures based on a transformer model.	Pretrained on large-scale datasets, including scanned documents, it demonstrates strong performance across various document-related tasks, such as form understanding and receipt processing. By combining textual data with the document layout, the model is able to capture the relationships between different sections, such as headers and footers, enhancing its ability to interpret the overall organization and context of the document.	Only uses text coordinates as visual information but does not fully exploit images or complex visual elements like logos, diagrams, or tables. Heavily reliant on accurate OCR to extract the text and corresponding coordinates, leading to performance degradation when OCR fails or the document has low text quality. Lacks features from raw image pixels, limiting its ability to understand non-textual or graphical elements in documents.

Approach	Key Idea	Observations
LayoutLMv2 [248]	Enhances LayoutLM by incorporating actual image features alongside text and layout, allowing the model to better handle images, logos, tables, and charts. Introduces spatial-aware self-attention mechanisms that capture both relationships between text blocks and their spatial positions, improving document layout understanding. Pretrained on multimodal tasks (text and images) to further enhance document understanding beyond pure text-based documents.	The addition of image features increases model complexity, resulting in higher memory and computational costs during training and inference. Although it adds visual features, it still relies on accurate OCR for text extraction. Errors in OCR can still negatively affect its performance.
LayoutLMv3 [249]	improving the model's understanding of both the visual and textual elements of documents. It uses sophisticated image feature extraction techniques, which allows the model to understand more complex visual elements. It reduces the dependency on OCR, making it more robust in handling documents	computationally intensive and requires significant hardware resources for training and inference due to the inclusion of advanced multimodal techniques. The increased complexity and size of the model may not be suitable for applications with limited computational resources, especially when real-time processing is required.
UDOP [250]	It pretrains the network using self-supervised tasks such as Layout Modeling, Visual Text Recognition, Joint Text-Layout Reconstruction, and Masked Image Reconstruction. These tasks help the model learn robust multimodal representations, which improve its performance across a variety of downstream tasks, including document classification, layout segmentation, information extraction (IE), question answering, and document natural language inference.	The approach is dependent on accurate text detection and OCR for its functionality. Furthermore, the method lacks an explanation or strategy for synthesizing and handling color documents, which poses a significant limitation when dealing with complex or multi-colored layouts. This absence of clarity in processing such documents highlights a gap in the approach's adaptability to more diverse and visually intricate document types.

Approach	Key Idea	Observations
XYLayoutLM [251]	The approach leverages a combination of text, visual, and layout features to enhance document understanding tasks. By utilizing the XY Cut algorithm, it ensures that the reading order of the document is accurately maintained, which significantly improves the system's performance. Additionally, the introduction of Dilated Conditional Position Encoding enables the system to effectively handle longer sentences, allowing for better processing of complex document structures while maintaining the integrity of both textual and visual information.	The approach heavily relies on the XY Cut algorithm, which struggles with documents that have complex layouts or various background color combinations, leading to potential inaccuracies in reading order. Additionally, the system's performance is sensitive to errors in OCR, as inaccuracies in text extraction can further degrade its ability to process and understand documents effectively. These limitations make it less reliable for documents with non-standard structures or visual complexities.

Approach	Key Idea	Observations
Self/semi-supervised		
SelfDocSeg [252]	The approach eliminates the need for manual labeling, relying solely on visual features without requiring OCR-based text extraction. Mask generation is straightforward and easy to implement, making the process efficient. The pretraining learning objective is formulated as a combination of cosine similarity and focal loss, enabling the model to learn robust representations from document images effectively.	The approach involves a complex training procedure, with its overall performance heavily dependent on the accuracy of the mask generation task. However, it faces challenges when applied to certain types of documents, such as those captured by a camera or color documents with varying shades of gray. In these cases, the method is likely to fail, as the generated masks may not accurately represent the document's structure or content.

2.4 Applications of Document Image Analysis

Researchers have created various applications to diminish the human effort required for document image analysis and assist individuals with disabilities. In this section, we delve into some of these works closely aligned with the applications we have developed, all centred around document image segmentation and classification.

2.4.1 Visual Assistance System

According to the World Health Organization (WHO), at least 2.2 billion people have a vision impairment or blindness worldwide. The computer vision community has undertaken significant efforts to deploy automated assistive systems for such people. Some of the unique assistive systems based on computer vision are — (i) vOICe[1] [253] vision technology is a sensory substitution system for blind people, which gives visual experience through live camera views by image-to-sound renderings, (ii) Electro-Neural Vision System (ENVS) [254] provides a virtual perception of the three-dimensional profile and color of the surroundings through pulses, (iii) a wearable travel aid for environment perception and navigation [255], (iv) a smartphone-based image captioning for the visually and hearing impaired people [256]. (v) Currency Recognition using Mobile Phones [257] (vi) Facebook's Automatic Alt-Text (AAT) [258] system is one of the most widely used assistive features, which automatically recognizes a set of 97 objects present in Facebook photos, (vii) FingerReader [259] is a wearable text-reading device to assist blind people and dyslexic readers. All these existing assistive systems fail to narrate properly the classroom slides for a VI student due to various logical regions present in the slides.

[1] https://www.seeingwithsound.com

2.4.2 Image-to-Image Translation

Researchers [260, 261] formulated image-to-image translation problems as pixel-wise classification or regression tasks. In all these formulations, the output space is treated as "unstructured" because each output pixel is conditionally independent of input image pixels [262]. On the other hand, conditional GANs learn a structured loss, which inspired many successful methods in image-to-image translation [262, 263].

Isola *et al.* [262] proposed a general-purpose solution for image-to-image translation, which required a significant number of labeled image pairs. Zhu *et al.* [263] proposed CycleGANs for image-to-image translation without labeled image pair. Yi *et al.* proposed dualGANs for general-purpose image-to-image translation using unpaired labeled images in [264].

2.4.3 Cross model image retrieval

Examining images of classroom slides has been an active area of research. Early works such as Deshpande et al. [265] showcased a real-time interactive virtual classroom multimedia distance learning system; Zhu et al. [266] presented a virtualized classroom project that emphasized automatic data collection, analysis, multimodal synchronization, compression, cross-media indexing, and archiving. Recent works like Haurilet et al. [7, 8] focused on layout segmentation for classroom slide images. Building on this, Jobin et al. [56] expanded the approach by implementing a narration system for classroom slides using the identified layout regions. In this section, we briefly present a literature survey on datasets and cross-modal retrieval in the space of lecture slides and outline the distinctions of our proposed dataset and approach.

2.4.4 Lecture Slide Retrieval Datasets

Several lecture slide datasets for the retrieval task have been introduced in the literature [59, 267–272]. Some of the popular datasets are listed here: (a) The LectureBank [268] comprises $1,352$ online lecture PDF files extracted from 60 Computer Science courses covering the following five sub-domains: Machine Learning, Natural Language Processing, Deep Learning, Artificial Intelligence, and Information Retrieval. Additionally, the dataset includes more than 1K concepts automatically extracted to form an in-domain vocabulary, along with annotations for prerequisite relation pairs that involve 208 concepts. (b) The VLEngagement dataset [267] includes content-based features, e.g., stop-word frequencies and video-specific features, e.g., silence and video duration, extracted from publicly accessible 4K scientific video lectures. The tasks related to this dataset are to predict context-agnostic engagement in video lectures and rank video lectures based on their engagement levels. (c) The ALV dataset [269] contains lecture videos that were generated using automatically created transcripts from academic online videos. (d) The LectureVideoDB [270] comprises 5K frames. These frames contain annotated text characters, with a focus on detecting and recognizing text within lecture videos. The videos cover a range of 24 distinct courses in science, management, and engineering. (e) The GoogleI/O dataset [271]

contains 209 presentation videos from the Google I/O conferences held between 2010 and 2012. The authors provide textual information from the spoken content and the slides in this dataset. (f) The LaRochelle dataset [272] comprises 47 French lecture recordings, totaling 65 hours of video, conducted in the author's lab. Each lecture, on average, features 50 slides. The researchers investigate cross-modal retrieval, employing a bag-of-words approach for both textual content and visual tokens. (g) The MLP Dataset [59] includes slides and spoken language, encompassing over 180 hours of video and more than 9000 slides, featuring contributions from 10 lecturers across diverse subjects such as computer science, dentistry, and biology.

	Features					Size
Datasets	Slide Segments	Sketches	Slide Text	Transcript	Summary	# Slides
VLEngagment [267]						
LectureBank [268]	✓(M)		✓(A)			51,939
ALV [269]	✓(A)			✓(A)		1,498
LectureVideoDB [270]	✓(M)		✓(M)			5,000
GoogleI/O [271]			✓(A)	✓(A)		
LaRochelle [272]	✓(A)		✓(A)	✓(A)		2,350
MLP Dataset [59]	✓(M)		✓(A)	✓(A)		9,031
LecSD Dataset (ours)	✓(M)	✓(M/A)	✓(A)		✓(M/A)	54,000

Table 2.2 Proposed LecSD Dataset as a comparison to the existing related datasets. The A and M represent that the features are extracted automatically and manually, respectively. Our dataset is larger with respect to the number of slide images and it has unique features of hand-drawn figure sketches and slide summaries as queries.

Most of these datasets retrieve slides or videos based on the corresponding transcript. However, in real-world scenarios, the query may differ from the transcripts. We assess our proposed dataset – LecSD in comparison with these datasets in Table 2.2. The baseline model for retrieving slide images given text and sketch query has been derived from the existing cross-modal image retrieval methods [59,273–275]. Of these models, CLIP [273] stands out as a popular benchmark for aligning images with text. CLIP undergoes training to excel in a diverse range of tasks, and this acquired task learning can be effectively harnessed through natural language prompting, enabling seamless zero-shot transfer to numerous existing datasets. Further, two successful approaches in this area are: (a) PCME [274]: which characterizes each modality by modeling probabilistic distributions in a shared embedding space, employing Hedged Instance Embeddings [276]. This approach captures the uncertainty associated with mapping an image to a latent embedding space by dispersing density across plausible locations. The embeddings are treated as random variables, and the model is trained based on the variational information bottleneck principle [277]. (b) PVSE [275]: which captures one-to-many alignment for cross-modal retrieval. It learns diverse representations for the data samples by fusing global context with locally-guided features through multi-headed self-attention and residual learning.

2.5 Discussion

This chapter provides an extensive account of the methodologies employed for feature extraction and document image segmentation, encompassing both pre-deep and deep learning periods. We briefly discuss various feature extraction and image segmentation techniques developed for document and general images. Features explicitly crafted for one classification task may not exhibit optimal performance in alternative tasks. Consequently, developing a versatile, all-encompassing feature holds considerable promise, particularly in the context of numerous document image analysis challenges.

Chapter 3

Deep Multi-modular Features

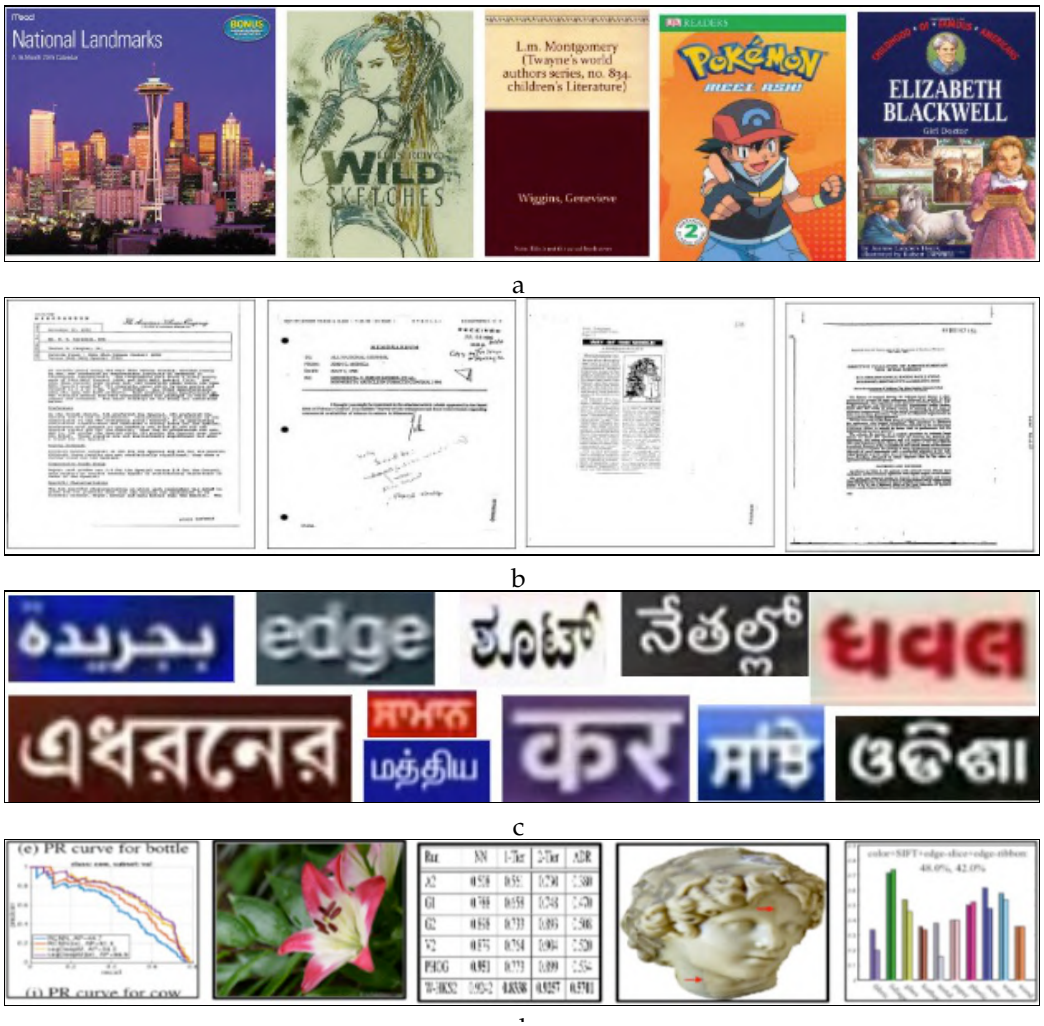
3.1 Introduction

Features play an essential role in various classification tasks related to document image analysis [44–47], such as document image classification [44, 45, 48–50], document figure classification [46], book cover classification [42], and script identification [47, 51, 52]. Researchers have proposed various approaches to tackle these classification tasks [278]. Each approach focuses on a specific problem, thereby failing to take cognizance of other problems. A Feature responsible for one particular task may not perform well on other tasks. We explore various deep features tailored for specific applications in the previous chapter. This chapter focuses on efficiently extracting general-purpose deep features from document images, which can be applied to various document tasks.

In script identification problems, characters or unique curves in a language are the key features for distinguishing it from other languages. Fig. 3.1(c) shows sample images of various languages. The texture features uniquely represent the curves and characters of a script to solve this problem [47,51,52]. On the other hand, most of the existing approaches to document image classification tasks focus on global shape features [44, 166, 167, 215] that capture the arrangement of various logical regions such as heading, paragraph, and figure. For example, the shape of documents with double-column formatting differs from documents with single-column formatting. Hence, shape features are more useful than texture features for solving this problem. However, both the texture and shape features are required to solve document figure classification [46] and book cover classification [42] tasks.

It is well established that the features corresponding to local image regions (patches) are more discriminative than the global features for various tasks [189]. These patches are called discriminative patches. Researchers designed various deep networks to extract discriminative features for fine-grained image classification tasks [189, 279]. These architectures extract only specific features (global, texture, or local discriminative) to solve a particular type of problem. Hence, the existing methods lack a generic model that extracts all the features discussed to solve the aforesaid tasks.

This work proposes a general-purpose deep network architecture to extract three types of features (i.e., global, texture, and local). We consider four different document image analysis tasks such as: (i) document image classification, (ii) genre classification, (iii) scientific document figure classifica-



a

b

c

Figure 3.1 Shows sample images from various considered datasets — (a) Book-Cover, (b) RVL-CDIP, (c) CVSI, and (d) `DocFigure`.

tion, and (iv) script identification. We choose these four tasks to show the effectiveness of our deep multi-modular feature on document image analysis, mainly the classification of the document images at various units. The document image and genre classifications act on whole document images. The document figures (e.g., Bar chart, Natural image, Pie chart, etc.) and words are smaller document units. The proposed approach performs various classification tasks by a judicial combination of these features. The extensive experiments on public benchmark datasets show that the proposed method outperforms the state-of-the-art on genre and document figure classification tasks and obtains comparable results on script identification tasks.

3.2 Deep Multi-modular Features

The traditional deep convolutional neural networks (CNNs) such as VGG-M [193], VGG-V [104], ResNet [106] and GoogleNet [194] learned the generic global features from the given input images. The learned global features may not always be capable of efficiently representing the input image's textural nature and enhancing the discriminative power of the networks [114]. As a result, these generic global features fail to classify sub-categories presented in an object category.

We propose a deep network shown in Fig. 3.2 capable of extracting global, discriminative local, textural features for document image classification. The proposed model consists of two blocks — (i) the first block consists of several convolution layers, and (ii) the second block consists of three different heads, mainly the global feature head, discriminative feature head, and encoding feature head. Each of the heads extracts a specific type of feature. The global feature head extracts the global feature. The discriminative feature head extracts the features corresponding to the discriminative local image patches. While the encoding feature head learns the encoded feature. Finally, the three types of extracted features are concatenated to represent the discriminative feature.

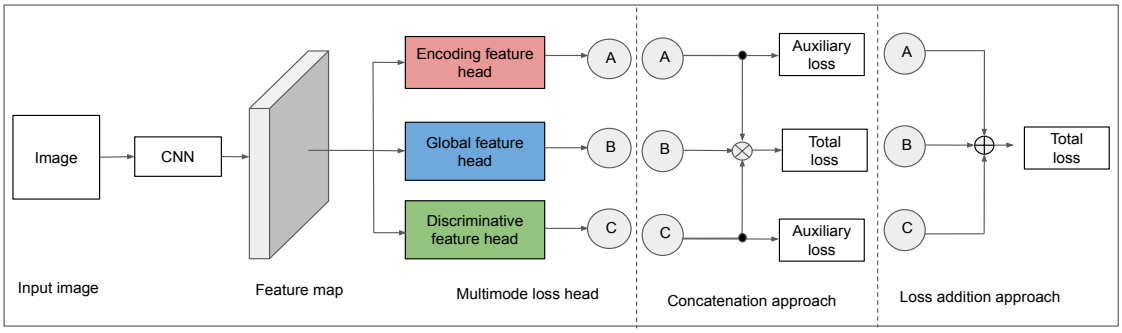


Figure 3.2 Presents a block diagram of the proposed approach. The input image passes through the layers of convolutional filters of an CNN architecture to extract convolutional features. From the convolutional feature, the model extracts three different modalities of features: an encoding feature, a global feature, and a discriminative feature. Here, $\otimes$ and $\oplus$ indicate concatenation and loss addition of features, and $\bullet$ represents sharing of the same features, respectively.

3.2.1 Encoding Feature Head

The texture feature extraction using deep learning proposed by Cimpoi *et al.* [114] lacks end-to-end training. Since, the model is not end-to-end trainable, it may not learn the discriminative texture features. In order to get discriminative texture features, we adapt the encoding network called Deep Texture Encoding Network (Deep TEN), proposed by Zhang *et al.* [184]. This network learns a codebook $C = \{c_1,\ c_2,\ c_3,\ ...,\ c_K\}$ containing K codewords, where $c_i \in \mathbb{R}^D$ and smoothing factor

$s = \{s_1,\ s_2,\ s_3,\ ...,\ s_K\}$ from a set of feature vector $X = \{x_1,\ x_2,\ x_3,\ ...,\ x_N\}$, where $x_i \in \mathbb{R}^D$. Irrespective of the number of feature vectors N, the encoder output is a fixed length representation $E = \{e_1,\ e_2,\ e_3,\ ...,\ e_K\}$, where $e_i \in \mathbb{R}^D$, and each e_k is calculated using Eq. (3.1).

$$e_k = \sum_{i=1}^{N} e_{ik} = \sum_{i=1}^{N} a_{ik} r_{ik}, \tag{3.1}$$

where r_{ik} is calculated by $r_{ik} = x_i - c_k$ and a_{ik} is the weight associated with each pair of x_i and c_k and it is calculated by Eq. (3.2).

$$a_{ik} = \frac{exp(-s_k||r_{ik}||^2)}{\sum_{j=1}^{K} exp(-s_j||r_{ij}||^2)} \tag{3.2}$$

The output of the encoder E is differentiable *w.r.t* the input X, codebook C and smoothing factor s. The codebook C and smoothing factor s are initialized with the random values within the range $(\frac{-1}{\sqrt{K \times D}}, \frac{1}{\sqrt{K \times D}})$ and $(\frac{-1}{\sqrt{K}}, \frac{1}{\sqrt{K}})$, respectively. The encoder is differentiable *w.r.t* the loss function and learn the codebook and smoothing factor in a supervised manner.

3.2.2 Discriminative Feature Head

We propose a discriminative feature head to learn features corresponding to the discriminative image patches, inspired by the work [189]. The discriminative feature head consists of an asymmetric two-stream architecture. The discriminative patch learning is a 1×1 convolutional layer followed by a Global Max Pooling (GMP) layer and a classification layer (fully-connected layer and a softmax layer). This architecture is not guaranteed to fire at discriminative patches as desired. A Cross-Channel Pooling (CCP) layer followed by a softmax layer is introduced to learn class-specific discriminative image patches.

The CCP layer is an average pooling layer. Instead of a fully connected classification layer, each classification output node's value is calculated by averaging the consecutive m values of the output of GMP layer. Hence, the number of output channels of the GMP layer should be $m \times L$, where L is the number of classes, and m is the number of filters per class. Since there are no learnable parameters in the cross-channel pooling layer, the 1×1 convolutional layer weights are adjusted directly via both loss functions of the classification layer and the softmax layer.

3.2.3 Global Feature Head and Backbone Neural Network

The global feature head consists of a convolutional layer, a fully connected layer, and a softmax layer. It captures the structural attributes and global nature of the input image. It is the same as the traditional image classification networks such as VGG-M [193], VGG-V [104], ResNet [106] and GoogleNet [194]. Like ResNet [106], we use a global average pooling layer before the fully connected layer. Hence, the

proposed architecture can handle arbitrarily shaped images. Fig. 3.3 shows the detail of the proposed architecture.

Discriminating head		Global head	Encoding head
		7×7, 64, stride 2	
		3×3 max pool, stride 2	
		$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128, C{=}32 \\ 1\times1, 256 \end{bmatrix} \times3$	
		$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256, C{=}32 \\ 1\times1, 512 \end{bmatrix} \times4$	
		$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512, C{=}32 \\ 1\times1, 1024 \end{bmatrix} \times6$	
	$1\times1, m\times L$	$\begin{bmatrix} 1\times1, 1024 \\ 3\times3, 1024, C{=}32 \\ 1\times1, 2048 \end{bmatrix} \times3$	$1\times1, 512$
CCP layer	max pool		Encoding layer
$1\times1, L$	$1\times1, L$		$L-d$ fc
		global average pool, $L-d$ fc	
		$4L-d$ fc , softmax	

Figure 3.3 Presents the detailed architecture of the proposed network. The thicker horizontal line represents a branching point where all the blocks touching a thicker horizontal line are connected.

3.2.4 Network Design

We have two design choices to combine the encoding, discriminative, and global feature heads. These are (i) adding the losses calculated with individual feature heads as suggested in [189] and (ii) concatenating all three feature heads followed by a linear layer to calculate the loss. In the first choice, the total loss (L_{total}) can be calculated using Eq. (3.3)

$$L_{total} = L_{enc} + L_{dis} + L_{glob}, \tag{3.3}$$

where the losses L_{enc}, L_{dis} and L_{glob} are calculate using the cross-entropy loss function as

$$\text{Loss}(y, \hat{y}) = -\frac{1}{N}\sum_{i=1}^{N}\sum_{l=1}^{L} y_{i,l} \cdot \log\left(\frac{\exp(\hat{y}_{i,l})}{\sum_{k=1}^{L}\exp(\hat{y}_{i,k})}\right). \tag{3.4}$$

Here, N is the number of samples in the batch, L is the number of classes, $y_{i,l}$ is the ground truth label for sample i and class l. $y_{i,l}$ is a one-hot vector, where the true class is 1, and others are 0. $\hat{y}_{i,l}$ is the predicted label for sample i and class l.

The direct sum of all the losses is a way to combine the various feature head. The contribution of multiple feature head losses depends on the numerous problems. For example, the encoding feature head loss is more relevant than the global feature head loss for a script identification in a word image. The second option of concatenating all the three feature heads, followed by a linear layer to calculate the loss. In this option, the linear layer learns the weight for each feature head based on the problem. The discriminative feature head fails to understand the discriminative patches for two reasons. These are — (i) the discriminative feature head consists of asymmetric two-stream architecture, and (ii) direct loss from the label is required as explained in Section 3.2.2 to learn the CCP layer.

To overcome these issues, we propose an auxiliary loss to learn the discriminative feature head as introduced in [238]. Similarly, we train the encoding feature head with an auxiliary loss to calculate between the labels and the prediction of the encoding head. Apart from the global feature head using softmax loss, another classifier is applied to learn the encoding and discriminative feature heads individually as an auxiliary loss. The additional loss helps optimize the learning process, while the master branch loss takes the most responsibility. Fig. 3.2 illustrates the detailed architecture of the proposed model.

3.2.5 Training Details

We train the proposed network using Stochastic Gradient Descent (SGD) with a momentum of 0.9 and a learning rate updated based on Cosine annealing strategy proposed in [280]. The initial learning rate η is set to 0.001 and after each epoch the learning rate is updated based on Eq. (3.5)

$$\eta_t = \frac{1}{2}\left(1 + cos\left(\frac{t\pi}{T}\right)\right)\eta. \tag{3.5}$$

The number of codewords, K in the encoder layer is set to 64 and the number of channels, m per class in the Cross-Channel pooling layer is set to 20. The size is set to 32 and the total number of iterations set to $100K$.

3.3 Experiments

We run all the experiments in an Intel i7 CPU processor with Nvidia GTX 1080Ti GPU, and our program utilizes 3259MiB of GPU memory and 2240MiB of CPU memory for the batch size of 32 during the training. We evaluate the performance of the proposed network architecture for four different tasks — (i) document image classification, (ii) genre classification, (iii) script identification, and (iii) scientific document figure classification.

3.3.1 Document Image Classification

In this task, our goal is to assign a pre-defined category label (like Advertisement, Email, Form, Letter, and Memo) to a given document image. It is often a prerequisite step towards high-level document image analysis tasks. Various types of document images contain distinct textural properties. More specifically, the different categories of document images can be discriminated against concerning the features corresponding to their local patches. This experiment combines global, discriminative local, and texture features to classify a given document into a pre-defined category.

3.3.1.1 Dataset and Pre-processing

We use the existing benchmark RVL-CDIP[1] [166] dataset to analyze the performance of the proposed approach on document image classification task. The dataset consists of scanned grayscale images of 16 categories of documents from lawsuits against American Tobacco companies. The dataset is divided into training, validation, and test sets, each containing 320K, 40K, and 40K images. The sample images of this dataset are shown in Fig. 3.1(b).

As discussed in [167], we apply similar pre-processing steps to the dataset. The steps are — (i) the images are resized to 384×384, and (ii) we duplicate a single image channel into three channels for network compatibility. Fig. 3.4 shows the challenges of intra-class dissimilarity and inter-class similarity present in the RVL-CDIP dataset. The first row of Fig. 3.4 highlights that different categories of documents (e.g., Advertisement, News article, Presentation, Scientific report, and Specification) have similar structural properties. While the second row of Fig. 3.4 highlights that documents of one particular category (e.g., Questionnaire) have distinct structural properties. High structural similarity among various classes of documents and high structural dissimilarity among document images of a specific category present in RVL-CDIP dataset makes document image classification tasks more complicated.

Table 3.1 The ablation study on the performance of document classification task under various settings, where $\oplus$ represents the addition of the losses separately, and $\otimes$ represents the concatenation of all in the last layers followed by a linear neural network.The number of codewords (k) and the number of channels per class in the CCP layer (m) are set to 16 and 20, respectively.

Feature	**Mode**	**Accuracy↑**
Global		91.18%
Global + Discriminative	$\otimes$	92.23%
Global + Encoding	$\otimes$	91.21%
Global + Encoding + Discriminative	$\otimes$	92.67%
Global + Discriminative	$\oplus$	92.44%
Global + Encoding	$\oplus$	91.83%
Global + Encoding + Discriminative	$\oplus$	**92.94%**

[1] http://www.cs.cmu.edu/~aharley/rvl-cdip/

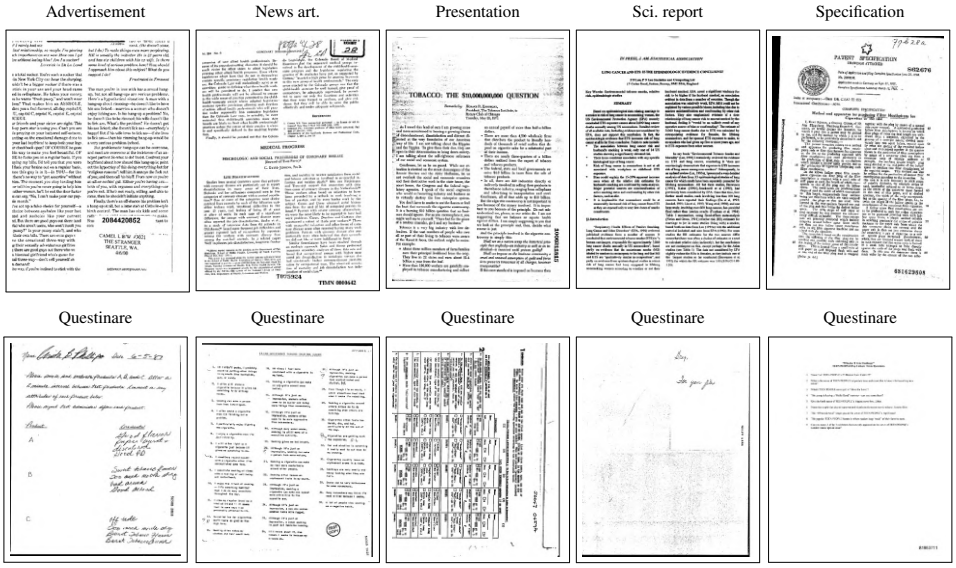


Figure 3.4 Shows challenges on the RVL-CDIP dataset. The first row indicates document images from different categories share similar structural properties. The second row highlights the document images from a specific type, namely "Questionnaire" with the separate structural property.

3.3.1.2 Ablation Study

Table 3.1 shows the ablation study to understand the contributions of various feature heads in document image classification tasks. The table also shows the improved performance of adding separate feature heads over the feature head's concatenation. The base network provides an accuracy of 91.18%. The concatenation of discriminating feature head to the global loss improved the accuracy by 1.05%. Adding the discriminative head to the global head improved the accuracy by 1.26%. The table also highlights that the addition of a discriminative feature head to the global feature head in both cases improves the accuracy (i.e., 92.23% and 92.44%)as compared to adding an encoding head to the global feature head (i.e., 91.21% and 91.83%). This result shows that the discriminating feature head is more informative than the encoding and texture features for the document image classification problem. However, the three feature heads' combination improves performance further in both cases (92.67% and 92.94%).

3.3.1.3 Learned Feature Visualization

We visualize the effects of the discriminative feature head shown in Fig. 3.3 for solving document image classification. For this purpose, we calculate the L_2 norm of the $1 \times 1, m \times L$ convolutional layer (shown in Fig. 3.3). We create a $2D$ heat map using the L_2 norm values to visualize the discriminative patches learned by the discriminative head. Similarly, we also visualize the L_2 norm calculated at the feature obtained after the $1 \times 1, 1024$ convolutional layer in the global feature head (shown in Fig. 3.3).

Fig. 3.5 shows the heat map, which is overlayed on the input image to visualize the learned discriminative patches and the global features of the input image. From Fig. 3.5, we observe that the

discriminative feature head concentrates on the meaningful discriminating regions of the image, which helps the network to differentiate one particular category of the document from other classes.

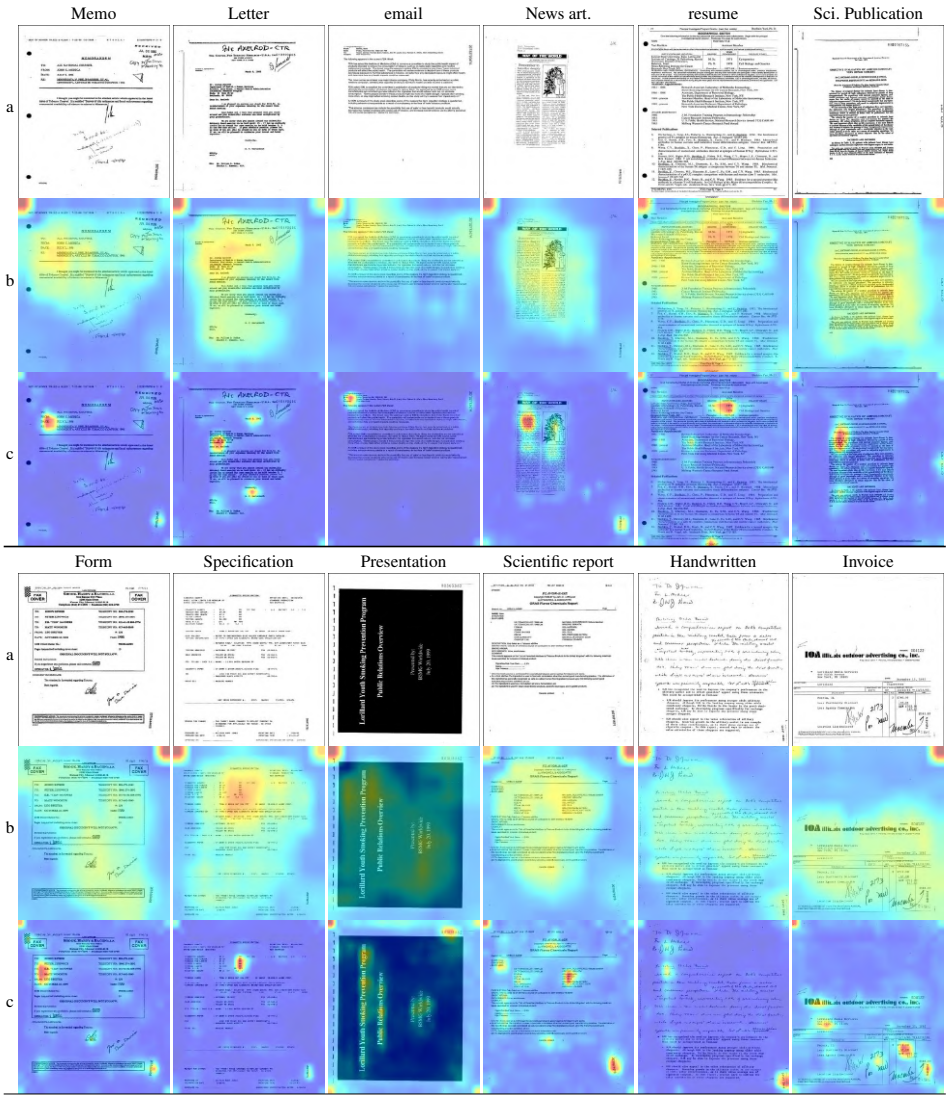


Figure 3.5 Visually shows the importance of global features vs. discriminative features for document image classification tasks. The 'a' rows show the sample images of RVL-CDIP dataset with category labels — Memo, Letter, Email, News article, Resume, Scientific Publication, Form, Specification, Presentation, Scientific report, Handwritten and Invoice. The 'b' rows illustrate the heat map calculated as the L_2 norm of the last convolutional layer before global average pooling, which is overlayed on the input image (i.e., global feature). The 'c' rows show the heat map calculated as the L_2 norm of the last convolutional layer before the discriminative feature head, which is overlayed on the input image (i.e., discriminative feature).

From Fig. 3.5, we observe that the global feature head gives importance to the top corners of the image irrespective of the category labels. While the discriminative feature head focuses on the meaning of fully discriminating regions, which is essential to discriminate one particular category of the document from other classes. For example, the discriminative feature head concentrates on the "address part" for both Memo and Letter categories of documents, while it focuses on "signature" only for Letters. In this case, "signature" is the discriminative feature to discriminate between Memo and Letter. The tabular structures generally occur in Form, Specification, Scientific report, and Invoice categories of documents and the discriminative feature head focuses on this tabular region (shown in Fig. 3.5). While the discriminative feature head fails to localize the handwriting region in the Handwritten category of documents. The encoding head overcomes this disadvantage by learning the textural pattern of the Handwritten type of documents. Figure 3.6 shows the confusion matrix obtained by the proposed approach. In the confusion matrix, the 'email' class performs with higher and lower accuracy for the class 'presentation' and 'form'. The most confusion is between 'sci. Report' and 'presentation'.

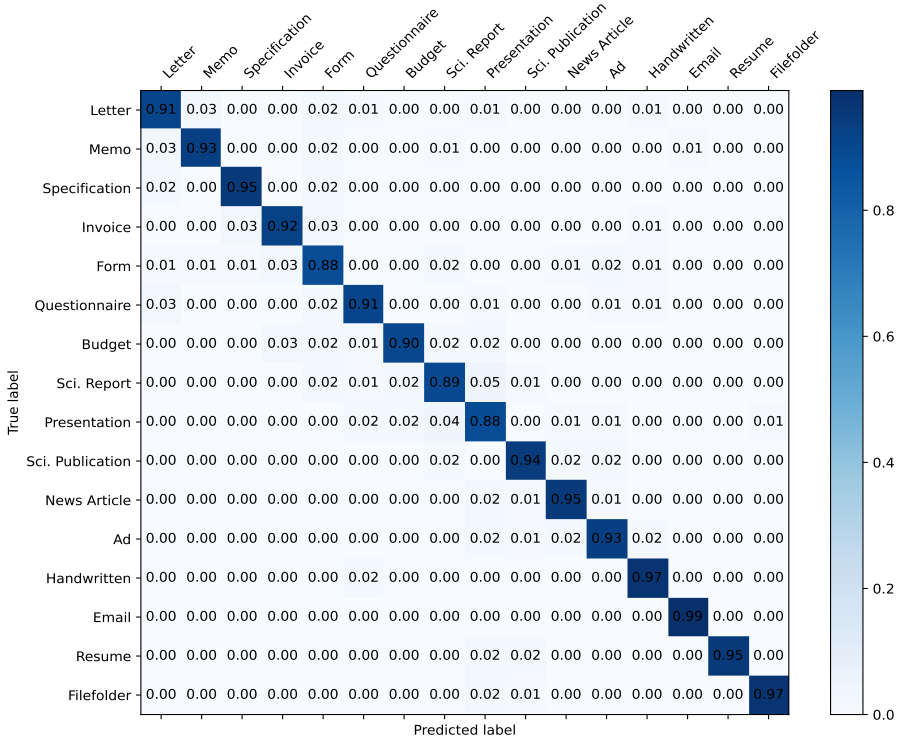


Figure 3.6 Shows the confusion matrix heat-map of RVL-CDIP dataset, obtained by the proposed approach

3.3.1.4 State-of-the-Art Comparison

Table 3.2 presents a detailed comparison between the proposed method and the recent methods for document image classification task on the RVL-CDIP [166] dataset. The works [44,45,166,167,215,217]

Table 3.2 Presents the document image classification accuracy on RVL-CDIP dataset, obtained by various methods based on the combination of images, text, and spatial features.

Method	**Accuracy(%)**
Methods based on only visual feature:	
Harley *et al.* [166]	89.80
SelfDoc [281]	90.49
Csurka *et al.* [215]	90.70
Tensmeyer *et al.* [167]	90.94
Afzal *et al.* [45]	90.97
Das *et al.* [44]	92.21
Sarkhel *et al.* [217]	92.77
Ours	**92.94**
Methods based on textual feature:	
BERT-base [219]	89.81
UniLMv2-base [282]	90.06
LayoutLMv1-base [218]	91.78
Methods based on textual + spatial features:	
UniLMv2-large [282]	90.20
LayoutLMv1-large [218]	91.90
Methods based on visual + textual + spatial features:	
Single Modal [220]	93.03
Ensemble [220]	93.07
SelfDoc [281]	93.81
LayoutLMv1-large [218]	94.43
LayoutLMv2-large [248]	95.65
DocFormer [221]	**96.17**

used only visual features for classifying document images. Among all existing methods using only visual features, Sarkhel *et al.* [217] obtained the highest accuracy (92.77%). Our method judicially combines visual features like global, textural, and local discriminative features for document classification and obtains an accuracy of 92.94% on the RVL-CDIP dataset. The proposed method obtains state-of-the-art performance while using only visual features.

Methods like BERT-base [219], UniLMv2-base [282], and LayoutLMv1-base [218] use only textual features for document classification. The LayoutLMv1-base [218] technique obtains the highest accuracy (91.78%) among all these methods. The proposed method using only visual features obtain 1.16% better accuracy than the LayoutLMv1-base [218] method using only textual features. Methods like UniLMv2-large [282] and LayoutLMv1-large [218] use both textual and spatial features for document classification. Among them, the LayoutLMv1-large [218] method obtains the highest accuracy (91.90%). Methods such as Single Modal [220], Ensemble [220], SelfDoc [281], LayoutLMv1-large [218], LayoutLMv2-large [248], and DocFormer [221] use visual, textual, and spatial features for document classification. Among all these methods, DocFormer [221] obtains the state-of-the-art performance (96.17% accuracy).

3.3.2 Book Cover Classification

It is the task of identifying the genre of the book from its cover image. It is one of the challenging tasks in document image analysis [42] because books come with a wide variety of covers and styles, including nondescript and misleading covers. Unlike other object detection and classification tasks, genres are not concretely defined. Another problem is the large number of books that make it unsuitable for exhaustive search methods.

3.3.2.1 Dataset and Pre-processing

We use the Book-Cover dataset [42], which contains 57K book cover images of 30 different genres. Each genre contains 1.9K images. Table 3.4 lists all the genre categories. Fig. 3.1(a) shows the sample images. This particular task is solved by the various global features extracted from deep neural networks such as LeNet [94] and AlexNet [165], and the results are reported in [42]. Even though our network takes input images of various sizes, we resize the input image to 227×227 to compare the result with the reported results of the existing approaches.

3.3.2.2 Ablation Study

Table 3.3 shows the ablation study on genre classification accuracy of various strategies. The table, shows that we obtain genre book classification accuracy of 35.10% only using the global feature. The use of the discriminative feature head and encoding feature head further improves the accuracy. The use of discriminative and encoding feature head along with global feature head enhance the accuracy by 0.52% and 0.14%, respectively, in the case of concatenation strategy. The combination of all the three

Table 3.3 The ablation study on the books' genre classification from their cover images in various settings, where $\oplus$ represents the addition the losses separately and $\otimes$ represents the concatenation of all in the last layers followed by a linear neural network. The number of codewords (k) and the number of channels per class in the CCP layer (m) are set to 32 and 20, respectively.

Feature	**Mode**	**Accuracy**$\uparrow$
Global		35.10%
Global + Discriminative	$\otimes$	35.62%
Global + Encoding	$\otimes$	35.24%
Global + Encoding + Discriminative	$\otimes$	35.82%
Global + Discriminative	$\oplus$	35.70%
Global + Encoding	$\oplus$	35.32%
Global + Encoding + Discriminative	$\oplus$	**36.17**%

feature heads further improves the result by 0.72%. As we expect, the addition of feature heads loss gives the best result (36.17%), which is 0.35% better than the concatenating feature head strategy. We also observe that the classification accuracy using discriminative and global feature heads is better than encoding and global feature heads.

3.3.2.3 Learned Feature Visualization

We visualize the L_2 norm of the learned global and discriminative features, shown in Fig. 3.7. The global feature highlights the background region containing the word "MAO", but the discriminative feature highlights the face region for the cover pages of the category Biographies. A human face is ubiquitous and appears on the cover pages of the books in the Biographies category. We observe that the discriminate feature head highlights the representative words of each category of book, which appear on the cover pages. A few examples are - the discriminative feature head focuses on the words "SAVE MORE" in Business, year "2016" in Calendar, "BISCUITS" in Cookbooks, "LAW" in LAW, and "PHARMACOLOGY" in Medical categories of books.

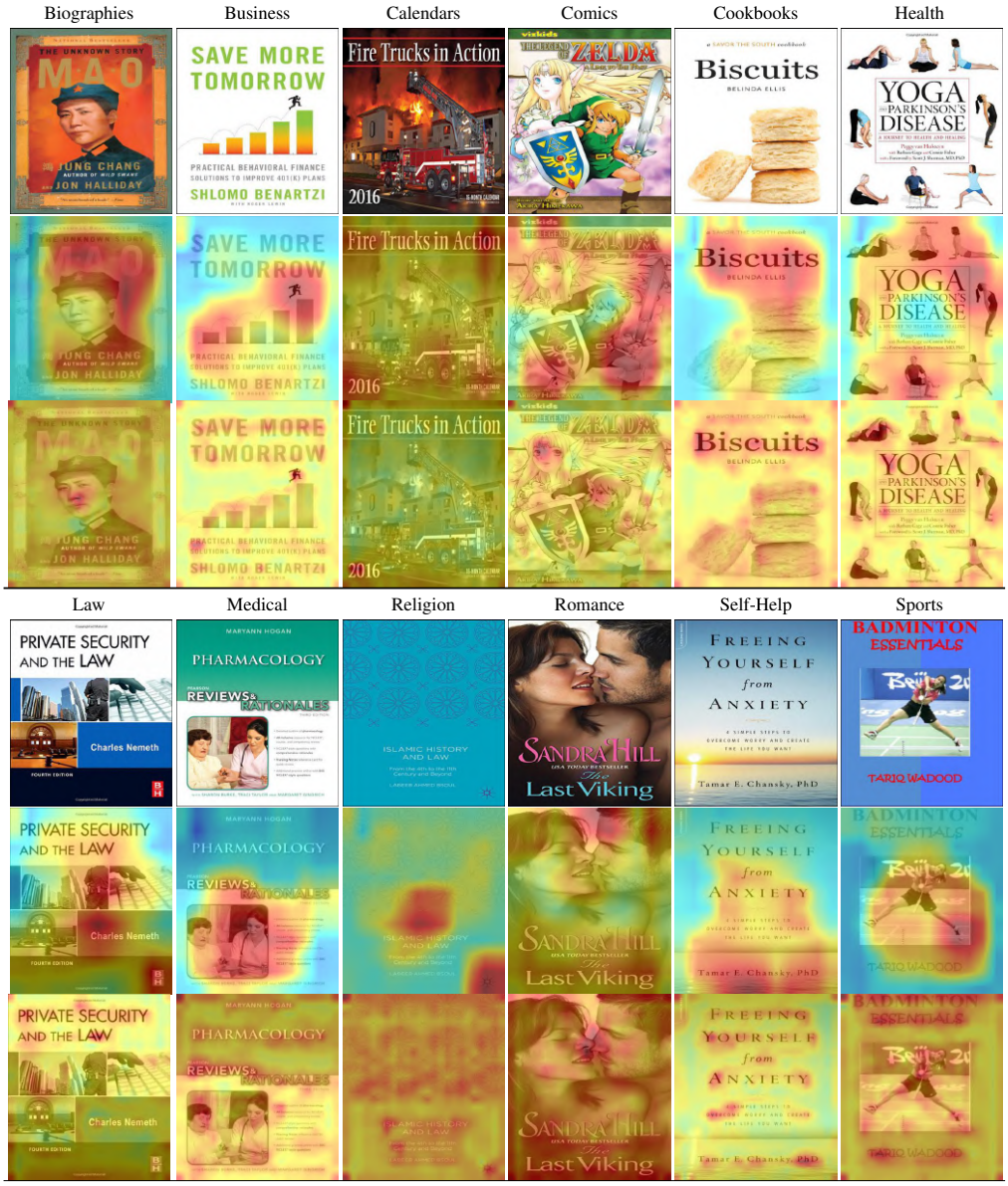


Figure 3.7 Shows the importance of global and discriminative features for genre classification tasks. The 'a' rows show the sample images of the Book-Cover dataset with category labels - Biographies, Business, Calendars, Comics, Cookbook, Health, Law, Medical, Religion, Romance, Self-Help and Sports. The 'b' rows illustrate the heat map calculated as the L_2 norm of the last convolutional layer before global average pooling, which is overlayed on the input image (i.e., global feature). The 'c' rows show the heat map calculated as the L_2 norm of the last convolutional layer before the discriminative feature head, which is overlayed on the input image (i.e., discriminative feature).

Table 3.4 Shows the class wise genre classification accuracy of Book-Cover dataset, obtained by various approaches.

Genre	**Classification Accuracy**↑			
	LeNet [42]	**AlexNet** [42]	**FV-CNN+ FC-CNN** [46]	**Our**
Arts & Photography	05.80	12.10	17.37	**27.37**
Biographies & Memoirs	05.30	13.20	17.37	**19.47**
Business & Money	10.00	12.60	16.84	**28.95**
Calendars	18.90	47.90	44.74	**71.58**
Children's Books	24.70	42.10	40.53	**44.74**
Comics & Graphic Novels	15.80	47.40	59.47	**67.89**
Computers & Technology	29.50	44.70	51.58	**55.79**
Cookbooks Food & Wine	14.20	43.70	47.37	**56.84**
Crafts Hobbies & Home	07.40	17.40	30.00	**40.53**
Christian Books & Bibles	08.40	07.40	13.16	**18.42**
Engineering & Transport	10.00	20.00	35.26	**36.84**
Health Fitness & Dieting	04.20	12.60	13.16	**18.95**
History	06.30	12.60	**25.79**	19.47
Humor & Entertainment	05.30	10.50	11.58	**18.95**
Law	14.70	25.30	35.79	**40.53**
Literature & Fiction	03.20	11.10	12.11	**17.37**
Medical Books	12.60	19.50	25.79	**34.74**
Mystery Thriller	23.70	34.20	36.84	**48.42**
Parenting & Relationships	14.70	24.20	30.53	**31.58**
Politics & Social Sciences	03.70	06.80	11.58	**13.68**
Reference	13.20	20.00	23.68	**28.42**
Religion & Spirituality	08.40	16.30	18.95	**29.47**
Romance	27.40	45.30	48.42	**56.32**
Science & Math	08.40	14.20	**24.21**	23.68
Science Fiction & Fantasy	14.70	35.80	14.74	**41.05**
Self-Help	13.70	14.20	**19.47**	15.79
Sports & Outdoors	05.30	14.70	**32.11**	**32.11**
Teen & Young Adult	07.90	12.10	15.79	**27.37**
Test Preparation	47.90	68.90	58.42	**73.68**
Travel	19.50	33.20	36.84	**45.16**
Total Average	13.50	24.70	29.00	**36.17**

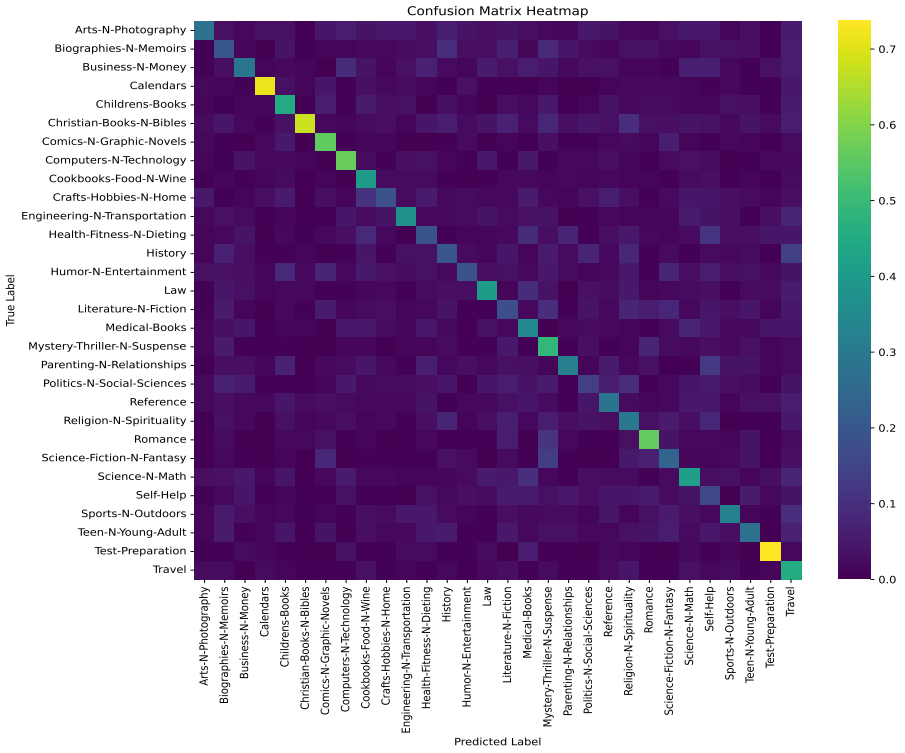


Figure 3.8 Shows the confusion matrix heat-map of Book-Cover dataset, obtained by the proposed approach.

3.3.2.4 State-of-the-Art Comparison

Table 3.4 presents a detailed comparison between the proposed and state-of-the-art approaches to book genre classification tasks. The proposed approach improves average accuracy by 22.67%, 11.47% and 7.17% over state-of-the-art methods — LeNet [42], AlexNet [42] and FC-CNN+FV-CNN [46], respectively. Compared to the proposed approach, FC-CNN+FV-CNN [46] feature outperforms in History, Science, Math, Self-help, & Sports and Outdoors classes. Iwana *et al.* [42] studied that the History category images have a high visual similarity with the images of other classes, such as Biographies & Memoirs, Politics & Social Sciences. The authors also pointed out that many miss-classifications occur in Biographies & Memoirs category. Figure 3.8 shows the heat-map of the confusion matrix obtained by the proposed approach.

3.3.3 Document Figure Classification

It is a task of assigning category labels like Block diagrams, Natural images, and Bar charts to the given document figure images. Classification of figures present in document images is complex due to inter-class visual similarity and intra-class visual dissimilarity. The existing methods [37, 53] using handcrafted features, fail to achieve good accuracy due to the extensive visual similarity among

subcategories. Recently, a few techniques [33, 34] convolutional neural networks have developed to solve this problem.

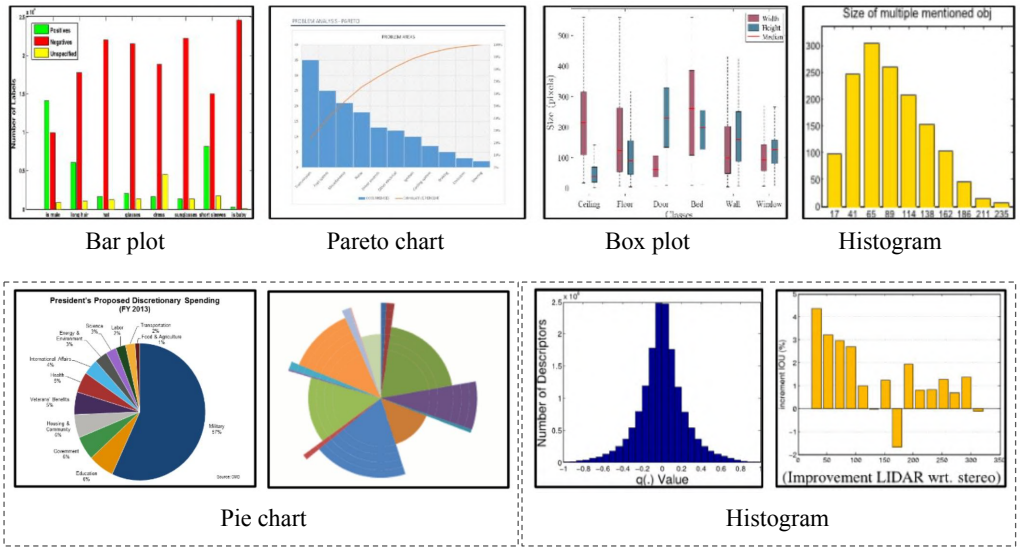


Figure 3.9 Shows challenges on the `DocFigure` dataset. The first row indicates images from different categories share similar visual properties. The second row highlights that the sample images from a specific type, like "Pie chart" and "Histogram" have distinct structural properties.

3.3.3.1 Dataset

We use the `DocFigure` dataset [46] for this particular experiment. The dataset contains $32K$ images of 28 different categories of figures, which are collected from scientific document images that correspond to scientific articles published in the CVPR, ECCV, and ICCV conferences in the last several years. The sample images are shown in Fig. 3.1(d). Jobin *et al.* reported results of three baselines — FC-CNN, FV-CNN, and FC-CNN+FV-CNN in [46]. The `DocFigure` dataset is the biggest dataset compared to the other document figure dataset Figureseer [34], Revision [37], Deepchart [33], and Karthikeyani and Nagarajan [36]. Fig. 1.3 illustrates the inter-similar and intra-dissimilar visual properties of the `DocFigure` dataset.

3.3.3.2 Ablation Study

Table 3.5 shows the ablation study for solving document figure classification tasks using various settings. Using the global feature alone gives an accuracy of 95.91% on the document figure classification task. It also shows that combining the discriminative feature with the global feature yields better results than combining the encoding feature with the global feature in concatenation and addition settings. Combining all three - discriminative, encoding, and global features gives the best classification accuracy (96.24% and 96.22%, respectively) for both scenarios.

Table 3.5 The ablation study on the document figure classification task in various settings, where $\oplus$ represents the addition of the losses separately, and $\otimes$ represents the concatenation of all the last layers followed by a linear neural network. The number of codewords (k) and the number of channels per class in the CCP layer (m) are set to 32 and 20, respectively.

Feature	Mode	Accuracy↑
Global		95.91%
Global + Discriminative	$\otimes$	96.13%
Global + Encoding	$\otimes$	96.05%
Global + Encoding + Discriminative	$\otimes$	96.22%
Global + Discriminative	$\oplus$	96.15%
Global + Encoding	$\oplus$	96.08%
Global + Encoding + Discriminative	$\oplus$	**96.24%**

3.3.3.3 Learned Feature Visualization

We visualize L_2 norms of global and discriminative features (shown in Fig. 3.10) to get an insight and analyze the importance of the feature for document figure classification tasks. From Fig. 3.10, we observe that the global feature head focuses on the complete object region while the discriminative feature head highlights only essential parts of the object region. These essential parts of the object region are vital for recognizing one category of figure from other categories. We noticed that discriminative feature head focuses on (i) a 3D-shaped region in a 3D Object, (ii) the trips of the bar in a Bar chart, (iii)a bubble circle in Bubble chart, (iv) lines in a Line graph, and (v)"India" region in Geographic map categories of document figures. Fig. 3.10 highlights the discriminative head learns features corresponding to the meaningful region of the input figure image.

3.3.3.4 State-of-the-Art Comparison

Table 3.6 shows the class-wise accuracy of the `DocFigure` dataset using various methods. The proposed method obtains improved accuracy over the state-of-the-art techniques — FC-CNN, FV-CNN, and FC-CNN+FV-CNN for 15 categories of document figures. The proposed method obtains the maximum accuracy improvement (i.e., 19.15%) of the Contour plot over state-of-the-art techniques. The proposed technique also improves average classification accuracy by 7.28%, 5.44%, and 3.34% over FC-CNN, FV-CNN, and FV-CNN+FC-CNN, respectively.

3.3.4 Script Identification in Multi-lingual Document Images

It is the task of identifying scripts in multi-lingual document images. It has a wide range of applications, including automatic storage of multi-script document images, document image retrieval, video indexing and retrieval, and document sorting in digital libraries [47]. Features play an important role in the script identification system. The article [47] summarizes the various feature categories

Table 3.6 Class-wise document figure classification accuracy of `DocFigure` dataset, obtained by various approaches.

Labels	**Classification Accuracy↑**			
	FC-CNN [46]	**FV-CNN** [46]	**FV-CNN+ FC-CNN**	**Our**
3D object	98.24%	94.73%	**98.53%**	97.81%
Algorithm	93.81%	91.75%	93.81%	**96.77%**
Bar chart	**93.97%**	91.97%	93.64%	93.10%
Box plot	91.39%	88.07%	**92.95%**	92.05%
Flow chart	92.53%	91.04%	91.38%	**97.01%**
Heat map	99.25%	95.89%	96.27%	**99.62%**
Histogram	**94.89%**	88.26%	**94.89%**	91.69%
Medical image	97.87%	92.55%	**98.93%**	96.90%
Pie chart	91.66%	89.81%	94.44%	**97.69%**
Polar chart	85.71%	78.57%	85.71%	**89.55%**
Area chart	84.61%	91.02%	92.30%	**100.00%**
Block diagram	97.26%	97.65%	**98.43%**	91.93%
Bubble chart	80.95%	91.66%	90.47%	**97.78%**
Confusion matrix	85.22%	89.65%	**93.10%**	91.36%
Contour plot	59.34%	74.72%	72.52%	**91.67%**
Geographic map	88.59%	95.81%	95.43%	**98.57%**
Line graph	98.49%	98.84%	**99.33%**	98.50%
Mask	**99.23%**	**99.23%**	**99.23%**	98.34%
Natural image	98.04%	98.25%	**99.23%**	98.57%
Pareto chart	87.17%	96.15%	**97.43%**	95.97%
Radar chart	78.94%	86.84%	85.52%	**90.00%**
Scatter plot	90.14%	91.19%	**93.66%**	89.89%
Sketches	95.65%	96.37%	**98.18%**	94.78%
Surface plot	76.76%	89.89%	88.88%	**93.59%**
Tables	97.25%	98.73%	97.67%	**99.34%**
Tree Diagram	67.04%	68.18%	70.45%	**82.14%**
Vector plot	79.86%	81.94%	86.80%	**94.32%**
Venn Diagram	87.03%	93.51%	93.05%	**96.00%**
Average Accuracy	88.96%	90.80%	92.90%	**96.24%**

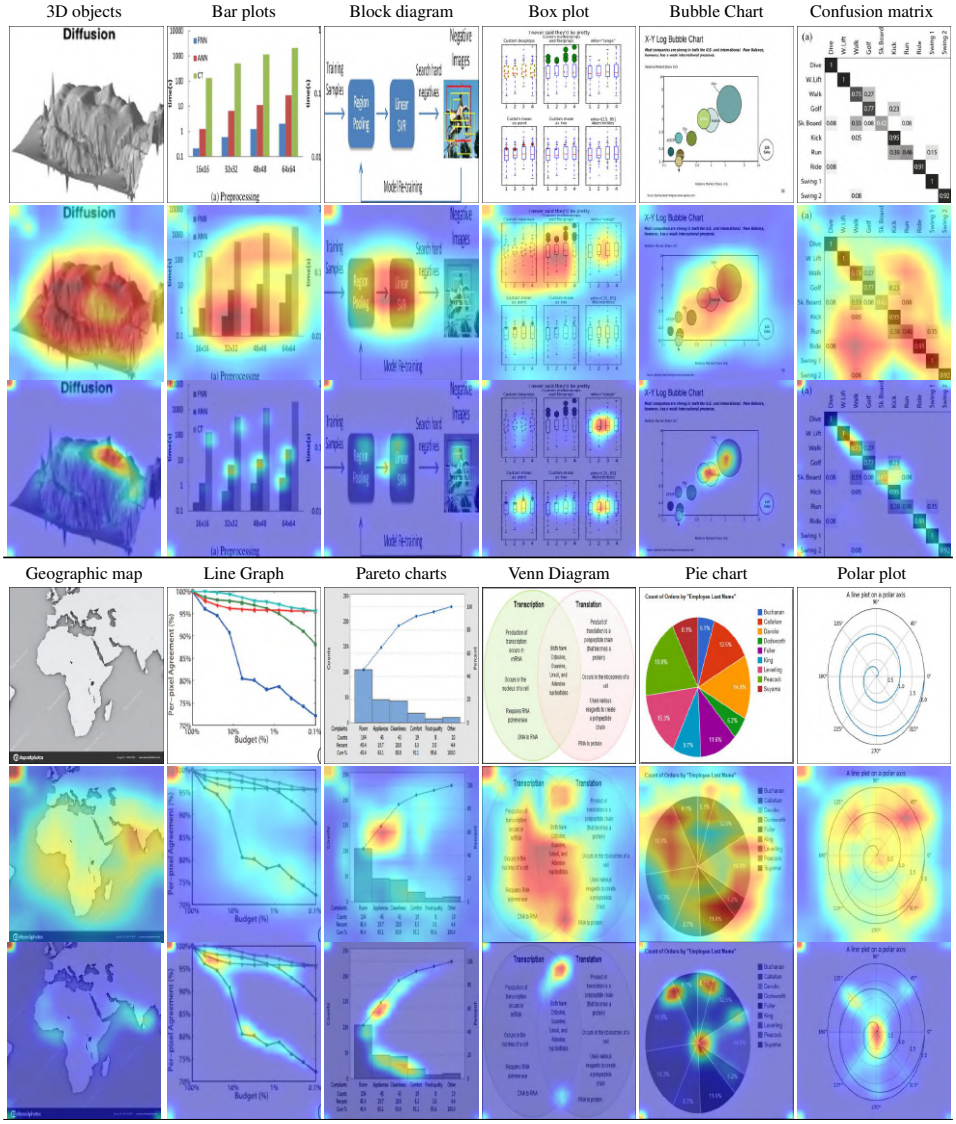


Figure 3.10 Shows the importance of global features vs. discriminative features for document figure classification tasks. The 'a' rows show the sample images of the `DocFigure` dataset [46] with selected category labels - 3D object, Bar chart, Block diagram, Box plot, Bubble chart, Confusion matrix, Geographic map, Line graph, Pareto chart, Venn diagram, Pie chart, and Polar chart. The 'b' rows illustrate the heat map calculated as the L_2 norm of the last convolutional layer before global average pooling, which is overlayed on the input image (i.e., global feature). The 'c' rows show the heat map calculated as the L_2 norm of the last convolutional layer before the discriminative feature head, overlaying on the input image (i.e., discriminative feature).

popularly applied in the script identification techniques. Various texture features like the Gabor filter, gray level co-occurrence matrix and wavelet are considered for script identification in multi-script documents [144, 155, 158–164]. Due to generalization capability, recently, deep features have been considered to identify scripts in multi-lingual document images [283].

Figure 3.11 Shows sample images of CVSI-2015 dataset. Each row of the image represent the sample images of (a) Arabic, (b) Bengali, (c) English (Roman), (d) Gujarathi, (e) Hindi (Devnagari), (f) Kannada, (g) Oriya, (h) Punjabi (Gurumukhi), (i) Tamil, and (j) Telegu, respectively.

3.3.4.1 Dataset

We use the CVSI-2015 dataset [284] for this particular experiment. The dataset is composed of images from news videos in various Indian languages. It contains 6412 training text images and 3207 test text images from 10 different scripts, namely Arabic, Bengali, English, Gujarati, Hindi, Kannada, Oriya, Punjabi, Tamil, and Telugu. The sample images of ten languages are shown in Fig. 3.11.

To handle the arbitrary size of the word image and enhance the unique curves in the script, we perform the following pre-processing steps: (i) conversion of all color images to grayscale images in which the character's areas are darker than the background, (ii) re-scale the image with widths 100, 40, 80 and 160, respectively, by keeping the aspect ratio constant, and (iii) arranging these scaled images in a 384×384 canvas as shown in the second-row and sixth-row in Fig. 3.12.

3.3.4.2 Ablation Study

Table 3.7 shows the ablation study on the script identification task under various network configurations. The global feature obtains 97.00% accuracy. In this particular problem, the encoded i.e. texture feature is more effective than the discriminative feature, as indicated in Table 3.7. The encoded feature combined with the global feature obtains 0.5% and 0.57% improved accuracy over the combination of discriminative and global features under concatenation and addition settings. The classification accuracy is further improved using a combination of global, discriminative and encoded features in both strategies.

Table 3.7 The ablation study on the performance of script identification task under various settings, where $\oplus$ represents the addition of the losses separately, and $\otimes$ represents the concatenation of all the last layers followed by a linear neural network. The number of codewords (k) and the number of channels per class in the CCP layer (m) are set to 8 and 20, respectively.

Features	**Mode**	**Accuracy↑**
Global		97.00%
Global + Discriminative	$\otimes$	97.71%
Global + Encoding	$\otimes$	98.21%
Global + Encoding + Discriminative	$\otimes$	98.64%
Global + Discriminative	$\oplus$	97.78%
Global + Encoding	$\oplus$	98.35%
Global + Encoding + Discriminative	$\oplus$	**98.83%**

3.3.4.3 Learned Feature Visualization

We visualize the heat map calculated as the L_2 norm of the last convolutional layer of global average pooling and the convolutional layer before the discriminative feature head, shown in Fig. 3.12. From the figure, we notice that the global feature head focuses on words of various scales. However, the discriminative feature head fails to learn a discriminative patch from the scrips, e.g., Arabic, Gujarati, Oriya, and Telugu.

Table 3.8 The script identification accuracy of CVSI-2015 dataset [284], obtained by various approaches.

Approach	**Accuracy↑**
C-DAC	84.66%
CUK	74.06%
HUST	96.69%
CVC-1	95.88%
CVC-2	96.00%
Shi *et al.* [52]	96.70%
Sing *et al.* [51]	98.13%
Our	98.83%
Google	**98.91%**

3.3.4.4 State-of-the-Art Comparison

We compare the performance of our method with state-of-the-art techniques on the script classification task. These results are summarized in Table 3.8. The table highlights that the proposed approach obtains a very close result to Google.

3.4 Ablation Study on Hyperparameters

The more effective hyperparameters in the proposed architecture are the number of codewords (K) in the encoding feature head and the number of channels per class in the CCP layer (m) of the discriminative feature head. We conduct a study to determine the optimum value of K and m for four classification problems.

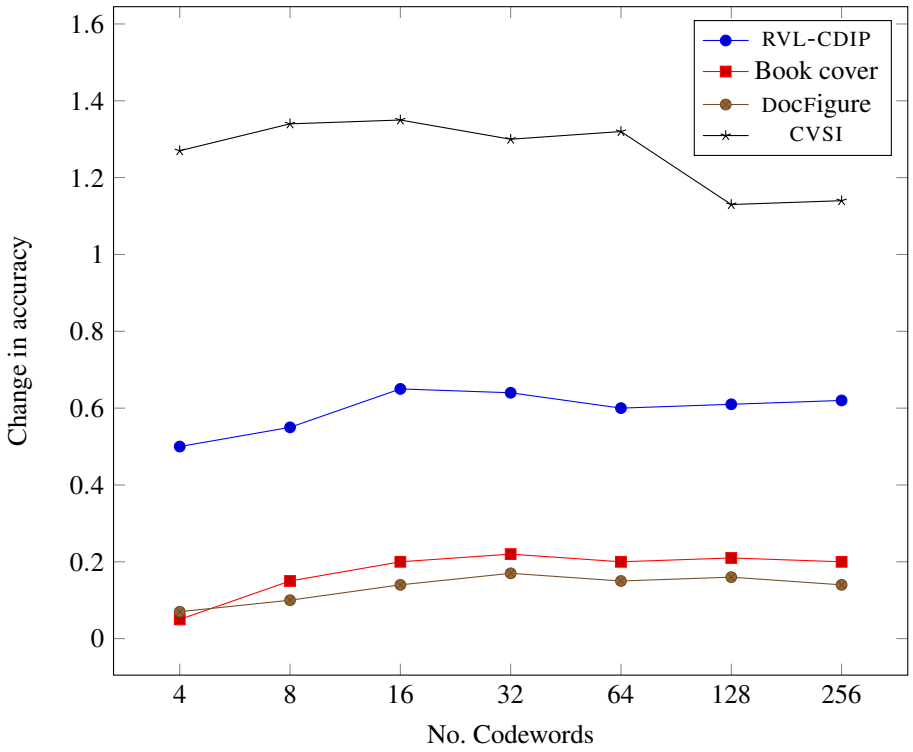


Figure 3.13 Shows the variation on accuracy with the changes in number of codewords K for four tasks — document image classification, book cover classification, document figure classification, and script classification corresponding to four datasets — RVL-CDIP, Book cover, `DocFigure`, and CVSI. Here, we combine the global feature head loss with the encoding feature head loss.

In the study of codewords (K), we use the architecture having global and encoding feature heads with the addition of the losses. We calculate the classification accuracy of four tasks - document image classification, book cover classification, document figure classification and script classification corresponding to four datasets — RVL-CDIP, Book cover, `DocFigure`, and CVSI with varying K from 4 to 256. Fig. 3.13 shows the variation in accuracy with the change of K. From the figure, we observe that the variation in accuracy is much less with the change in K for all tasks (almost flat curve). We also observe from the figure that the best value of K for each dataset directly relates to the total number of classes present in the dataset. The CVSI dataset has 10 classes and the best K is 8 for this dataset. The

RVL-CDIP dataset has 16 classes and the best K is 16. The Book cover and `DocFigure` datasets have 30 and 28 classes, respectively, the best K is 32 for both datasets.

In the study of the number of channels per class in the CCP layer (m), we use the architecture with global and discriminative feature heads with the addition of the losses. First, we calculate the classification accuracy of four classification tasks with m value ranging from 5 to 35. Fig. 3.14 shows the variation on accuracy with the changing m for four different tasks corresponding to four datasets. From the figure, we observe that the best accuracy is obtained when the value of m is equal to 20, irrespective of the dataset.

3.5 Discussion

We introduce a deep multi-modular feature extraction architecture for various classification tasks in document image analysis. The proposed architecture extracts three features - discriminative, encoded/texture, and global. Diverse experiments conclude that a combination of discriminative, texture, and global features performs better for various classification tasks - document image classification, genre classification of book, document figure classification, and script identification. We visualize the L_2 norm of the learned global and discriminative features in the form of the heat map, highlighting their importance for various classification tasks. The heat maps highlight that discriminative features are more useful for document image classification, document figure classification, and genre classification than other features. In contrast, the encoded feature is more essential than other features for the script identification task. In the future, we will explore the proposed architecture for classroom slide retrieval and signature verification.

Figure 3.12 Shows the importance of global vs. discriminative features for document figure classification tasks. The 'a' rows show the sample images of the CVSI dataset [284] with languages — Arabic, Bengali, English, Gujarati, Hindi, Kannada, Oriya, Punjabi, Tamil, and Telugu. The 'b' rows show the pre-processed word image before being fed into the classification network. The 'c' rows illustrate the heat map calculated as the L_2 norm of the last convolutional layer before global average pooling, which is overlayed on the input image (i.e., global feature). The 'd' rows show the heat map calculated as the L_2 norm of the last convolutional layer before the discriminative feature head overlayed on the input image (i.e., discriminative feature).

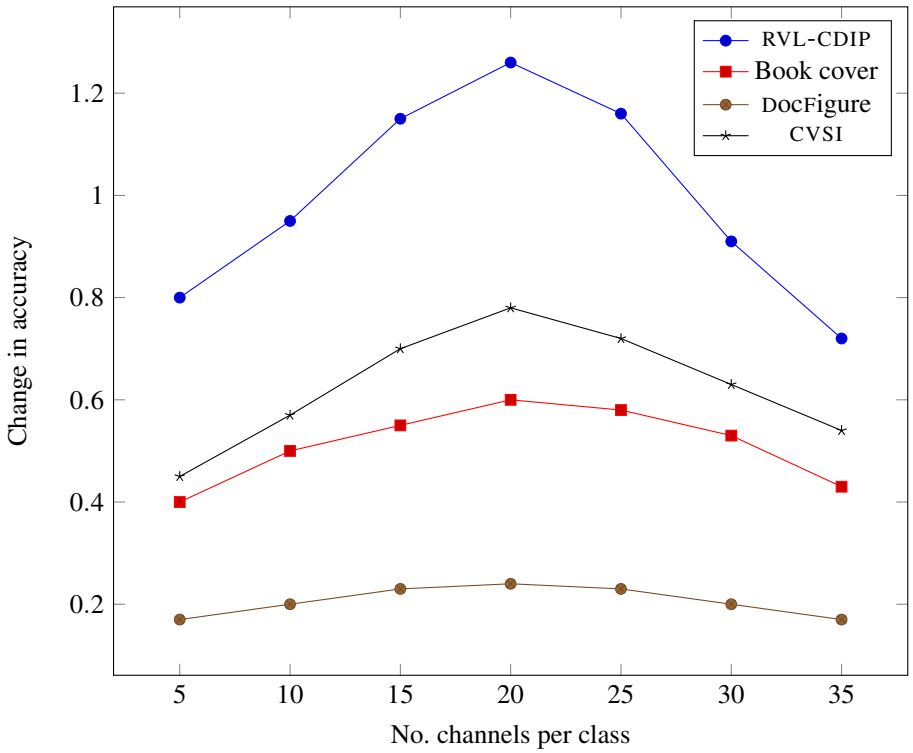


Figure 3.14 Shows the variation in accuracy with the changes in the number of channels per class in the CCP layer (m) for four tasks - document image classification, book cover classification, document figure classification and script classification corresponding to four datasets - RVL-CDIP, Book cover, DocFigure, and CVSI. Here, we combine the global feature head loss with the discriminating feature head loss.

Chapter 4

Document Image Segmentation

4.1 Introduction

In the last chapter, we analyze deep features and their effectiveness in various document classification tasks. Following classification, we categorize documents into specific types, such as scientific papers, emails, forms, and handwritten documents, each with its own logical regions. For instance, equations are typically found in scientific documents but not in news articles. As discussed in the Chapter 1 introduction, identifying logical regions is a crucial step in any document understanding system. This chapter focuses on segmenting three types of document images: scientific documents, classroom slide images, and historical document images.

Document image layout segmentation is pivotal in unlocking the wealth of information encapsulated within diverse forms of documents, spanning historical manuscripts, scientific papers, to modern educational resources like classroom slide presentations. These documents, each with unique characteristics, pose distinct challenges and opportunities for layout segmentation techniques. This introductory section embarks on an exploratory journey to illuminate the significance of layout segmentation in the context of these varied document types.

Historical document images, often aged and fragile, represent invaluable records. They offer insights into the cultures, languages, and knowledge of bygone eras. However, their aged nature can introduce intricate layouts, scripts, and visual artifacts that challenge traditional segmentation methods. The successful segmentation of historical documents is vital for unlocking the troves of historical knowledge they contain.

In the realm of scientific documents, we encounter a vastly different landscape. Scientific papers are the lifeblood of the academic world, encapsulating groundbreaking discoveries and research findings. However, they present their unique challenges in layout segmentation. From multi-column layouts to intricate figures, tables, and mathematical notations, scientific documents' structured but intricate organization requires tailored segmentation approaches to facilitate information retrieval and content analysis.

Educational materials, in the form of classroom slide presentations, have also emerged as a prominent source of knowledge dissemination. As educational technology advances, these materials often

integrate multimedia content, text, and images. Efficient layout segmentation is crucial for improving accessibility, aiding content navigation, and enhancing the understanding of complex subjects, particularly for students with disabilities.

This exploration of document image layout segmentation will delve into the challenges and advancements specific to these domains, unravelling the methods and techniques that enable us to transform these documents into valuable resources for research, education, and preserving our historical heritage.

4.2 Historical document image segmentation

Document image segmentation can be considered the primary stage of document image analysis (DIA) system. The objective of this module is often to segment the document image into semantically similar regions such as text, graphics, comments, decorations, backgrounds, etc. These segmented regions are processed and analyzed separately in DIA for a meaningful understanding of the content of the document. Thus, the segmentation module enables the DIA system to handle the document images having complex layouts.

Compared to the well-structured printed document images, the document segmentation problem is more challenging for historical handwritten documents. Hence, analysing and understanding of historical document images is an active area of research. Challenges of historical handwritten document images, such as unstructured layouts, degradation, various handwritten styles, etc., are exemplified in Figure 4.1.

The early approaches for document segmentation (such as [285]) are based on binarization and connected component analysis. This approach fails with images that have non-uniform background colors (Figure 4.1(b)). To get rid of this issue researchers started to classify image patches or superpixels to segment the image instead of taking connected components. Various features such as color and texture [134], SIFT, SURF, LBP, HOG, etc., are extracted for segmentation.

Inspired by the success of deep features over the hand-crafted features in the recent past [114], in this article, we explore the effectiveness of deep features in document segmentation. Document images have many practical difficulties in using CNNs directly. The main difficulty is that the input image should be resized to fit the size of the input of the CNN architecture. It may badly affect the performance of the method. Moreover, the time and data required to train deep neural networks are larger than those required for standard statistical machine learning methods such as Support Vector Machines (SVM). This article investigates how to utilize the non-linear filters of a pre-trained deep CNN without fine turning for the document segmentation.

In our study, we choose the deep features FC-CNN and FV-CNN, which produce the state-of-the-art results in texture, material and scene recognition [114]. Different text in the document has different texture properties. The results of our study conclude that these deep features are capable of distinguishing the texture of different writing styles (text and comments) and decorations very efficiently. To study the

effectiveness of the deep feature for historical document images, we visualize the deep features using the t-SNE [113] method.

In this article, we experimented on six different historical handwritten challenging segmentation datasets. The sample images from these datasets are shown in figure 4.1, and details can be found in Table 4.1. G. Washington, Parzival, and St. Gall document images are from the IAM historical document database.[1] A new dataset with a more complex layout introduced [286] are CB55, CSG18 and CSG863.

We pose the document image segmentation task as a semantic segmentation problem. Semantic segmentation is a popular problem in computer vision in which each pixel is assigned to its most appropriate label from a predefined label set. Many approaches that use CNN for semantic segmentation [287]. In this work, we first extract deep features from document image pixels and train an SVM classifier to label the pixels. The performance of the proposed approach is compared with the following approaches: (i) local MLP [198], (ii) CRF [288] and (iii) CNN [289] on six different historical document image datasets. The experimental studies conclude that the proposed method gives superior results consistently.

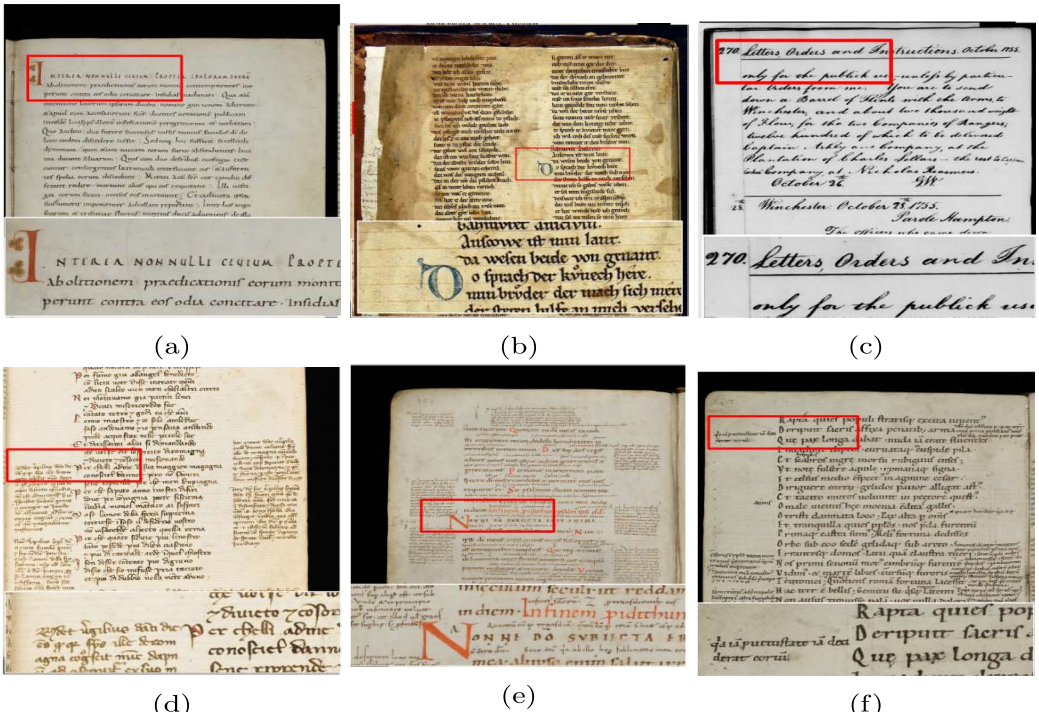


Figure 4.1 Sample historical handwritten document images for layout segmentation from the following datasets: (a) St. Gall, (b) Parzival, (c) G. Washington, (d) CB55, (e) CSG18, (f) CSG863. The bottom part of each image is the zoomed view of red rectangular region.

[1] http://www.fki.inf.unibe.ch/databases/iam-historical-document-database

4.2.1 Proposed Method

We pose the document image segmentation problem as a pixel labeling problem. Consider a document image I of size $w \times h \times d$, where w, h and d are the width, height and the number of color channel of the image respectively. Let $x_{i,j}$ is the pixel of image I at position (i, j), where $i \in \{1, \ldots, w\}$ and $j \in \{1, \ldots, h\}$. We train a statistical model which learns the label l from a set of labels L for each pixel $x_{i,j}$. The label set $\mathcal{L}$ for historical document images is set as $L = \{$body text, comments, decoration, background$\}$ similar to the past methods. We extract deep features proposed by Cimpoi *et al.* [114] from image patch surrounding to each of pixel x_{ij} in document image I. Finally, using an SVM classifier, on top of the extracted features, a label l to each pixel x_{ij} is assigned.

4.2.1.1 Image pre-processing

In our approach, we apply the SLIC superpixel segmentation algorithm. In document images, most of the pixels in a superpixel share the same label. The superiority of the superpixel based labeling approach over the pixel labeling approach for the page segmentation task has been demonstrated in [134].

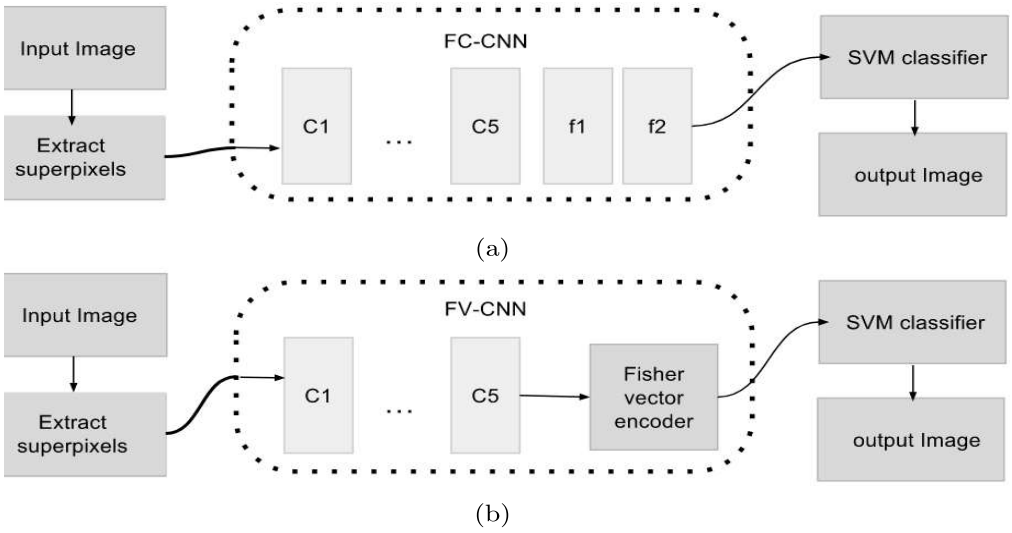


Figure 4.2 The proposed approaches: (a) and (b) are separate pipelines for segmenting a document image using FC-CNN and FV-CNN features. The dotted box represents the feature extraction module, c1, ..., c5 represent convolutional layers of CNN and f1 and f2 represent the fully connected layers of CNN.

4.2.1.2 Texture of document images

In early works such as [293], the texture is characterized by the arrangement of local patterns by the distribution of local filter bank responses. The filter banks can capture edges, spots, and bars at different scales and orientations. The work [294] proposes *textons* which are defined by combining the

Table 4.1 Details of of the dataset used in our experiments. TR, TE, and VA denote size of the training, test, and validation sets respectively.

Dataset	Image size (pixels)	TR	TE	VA
G. Washington [290]	2200×3400	10	5	4
St. Gall [291]	1664×2496	20	30	10
Parzival [292]	2000×3008	20	30	10
CB55 [286]	4872×6496	20	13	2
CSG18 [286]	3328×4992	20	10	10
CSG863 [286]	3328×4992	20	10	10

filter responses. The idea of *textons* is improved by new pooling schemes such as soft-assignment [295] and Fisher Vectors (FVs) [296]. The work [67] uses FV encoding on the SIFT features, which achieved state-of-the-art results in textures, objects and scene detection tasks.

One of the key observations made while visualizing the weights of a learned CNN [297] is that the initial layer learns to detect low-level patterns such as dots, line edges, strokes, etc. While later layers in the network learn the higher-level structures from images like faces, the shape of an object, etc. Even if the CNN is trained with a different dataset or non-document images, the filters are capable of emphasizing various texture patterns in an image.

This article proposes two types of deep features, namely FC-CNN and FV-CNN, inspired from [114]. Both descriptors are based on the same CNN features [103] obtained from an off-the-shelf CNN pre-trained on the ILSVRC 2012 dataset [110]. We evaluate the performance of segmentation using both features.

The FC-CNN descriptor is obtained by extracting the output of the penultimate fully connected layer of a CNN, including the nonlinear gating function, applied to an input image. This feature can be considered as an object descriptor because a fully connected layer captures the overall shape of an object. However, it has some drawbacks in the case of document images: i) since it is using a fully connected neural network, the input image patch should be resized, which makes the feature faulty ii) the FC-CNN feature represents the entire shape of an object rather than the texture. However, in document images, there is no importance in the shape of the object in a patch because it may be very random.

The FV-CNN feature overcomes the above disadvantages because the feature is extracted by FV pooling of the convolutional filter response rather than the fully connected layer. Hence, the FV-CNN is a more efficient way to describe the texture of an image than the FC-CNN. Since the feature responses are taken from the convolutional layer, no input patch/image rescaling is required. In the work [114], it is proved empirically that the efficiency of the FV-CNN feature improves from the initial convolutional layer to the final convolutional layer monotonically. Hence, in all our experiments, we use the final convolutional layer of the pre-trained neural network.

4.2.1.3 Document image segmentation

First, from the given training data, we generate superpixels by running the SLIC algorithm using an appropriate region size and regularization parameters. To get the contextual information about a superpixel, we crop each superpixel into patches of size $p \times p$ with the center aligned to the center of the superpixel. The optimum patch size p is calculated by minimizing the superpixel classification error (Figure 4.5). Then we extract the deep features for each superpixel region from the corresponding patches.

In FV-CNN, from the densely pooled response of the input image in the convolutional layer, we learn a Fisher Vector (FV) encoder with 64 Gaussian components. This encoder is used for creating the FV-CNN features from the densely pooled response of the convolutional layer of pre-trained CNN architecture. We use VGG-M [193] architecture trained on the ILSVRC 2012 dataset in all our experiments. The feature dimension in FV-CNN depends on the number of components of the Fisher Vector encoder and the number of filters in the convolutional layer. However, the dimension of FC-CNN is the same as the dimension of the fully connected layer.

We train an SVM classifier with predefined labels to model the extracted features. In the testing stage, first we extract features from each superpixel. Then, we classify the feature and label the superpixel region in the output image with the corresponding label of the superpixel. The proposed approaches using features FC-CNN and FV-CNN are described in Figure 4.2.

4.2.2 Dataset

We consider various datasets to demonstrate the effectiveness of the proposed algorithm on different kinds of document images.

The George Washington dataset is the manuscript written by George Washington and his associates in the year 1755. This dataset is very neat and less complicated, and it has only two regions, namely text and background.

The Parzival dataset manuscript contains the epic poem Parzival, one of the most important epic works in the European Middle Ages and was written with ink on parchment in the Middle High German language in the year of 13th-century. The entire document is written in the two-column format and the background color fades and ink spreads are present in the dataset. Currently, the author of the manuscript is unknown, but researchers are expecting that more than one author wrote the entire dataset.

The St. Gall dataset is the manuscript describing the hagiography of St. Gall by Walafrid Strabo. This is the oldest dataset used in our experiment, and the manuscript was created in the 9th-century. The Latin manuscript is written by a (probably) single experienced hand in Carolingian script with ink on parchment. It is a single-column document with predominantly Carolingian minuscules and fewer upper script letters that emphasize the structure of the text and some richly ornamented initials. The Carolingian minuscules were written in black ink, but the upper script letters and the richly ornamented initials were written in an unstructured way with different colors.

The very recent and challenging historical document image layout segmentation database is the DIVA-HisDB [286]. The dataset consists of 50 pages of three different medieval manuscripts with challenging layouts, namely CB55, CSG18, and CSG863. The three datasets represent the special class of manuscript layout, which have the main text column and marginal notes and /or interlinear glosses, additions, and corrections, as shown in figure 4.1. The CSG18 and CSG863, dating from the 11th century, were written in Latin language using the Carolingian minuscule script, but the writer is unknown. The CB55 manuscript is written by a single author from the 14th century and that shows a different script (chancery script), language (Italian and Latin), and layout.

4.2.3 Experiments

We use the standard evaluation scheme to analyze the performance of the proposed method. Since document image segmentation can be considered a semantic segmentation task, for comparisons, we use pixel accuracy and region of intersection over union(IoU) [26] as the evaluation metrics. Let n_{ij} and n_{cl} be the number of pixels of class i predicted to belong to class j and the total number of classes, respectively. Let $t_i = \sum_j n_{ij}$ is the total number of pixels of class i. The following are the metrics we use to evaluate the segmentation:

- pixel accuracy: $\sum_i n_{ii} / \sum_i t_i$
- mean accuracy: $(1/n_{cl}) \sum_i n_{ii} / \sum_i t_i$
- mean IoU: $(1/n_{cl}) \sum_i n_{ii} / (\sum_i t_i + \sum_j n_{ji} - n_{ii})$
- frequency weighted IoU:
  $(\sum_k t_k)^{-1} \sum_i t_i n_{ii} / (\sum_i t_i + \sum_j n_{ji} - n_{ii})$

4.2.3.1 Evaluation on historical document images

To evaluate the proposed method, we use six different historical handwritten document datasets In the SLIC superpixel extraction stage, we chose 10 pixels as the region size and the regularizing parameter as 0.01.

The quantitative results are presented in Table 4.2, and the qualitative results are shown in Figure 4.3. From Table 4.2, it is seen that the proposed method with the FV-CNN feature achieves a maximum of 9%, 19%, 19%, and 14% improvements over the CNN method [289] in pixel accuracy, mean accuracy, mean IoU, and frequency-weighted IoU, respectively. However, the recent approaches like SegNET, Deeplabv3 [300], MT-FCN [299], dhSegment [298] outperformed the proposed approach. These approaches utilize the fully convolution network-based architecture with larger contextual information than the proposed approach.

Table 4.2 Comparison on results of historical document image segmentation on the six different dataset with Local MLP, CRF, CNN and the proposed methods FC-CNN and FV-CNN.Uses the patch size of 30, we compare our approach with the latest result from dhSegment [298], MT-FCN [299], Deeplabv3 and SegNet [300]

Datasets	G. Washington				Parzival				St.Gall			
Evaluation	pixel acc.	mean acc.	mean IoU.	f.w. IoU.	pixel acc.	mean acc.	mean IoU.	f.w. IoU.	pixel acc.	mean acc.	mean IoU.	f.w. IoU.
Local MLP [198]	87	89	75	83	91	64	58	86	95	89	84	92
CRF [288]	91	90	76	85	93	70	63	88	97	88	84	94
CNN [289]	91	91	77	86	94	75	68	89	98	90	87	96
FC-CNN (ours)	94	92	80	89	**97**	74	**71**	**94**	**99**	**91**	**88**	**98**
FV-CNN (ours)	**95**	**93**	**81**	**91**	**97**	**76**	**71**	**94**	**99**	**91**	**88**	**98**
Datasets	CB55				CSG18				CSG863			
Evaluation	pixel acc.	mean acc.	mean IoU.	f.w. IoU.	pixel acc.	mean acc.	mean IoU.	f.w. IoU.	pixel acc.	mean acc.	mean IoU.	f.w. IoU.
Local MLP [198]	83	53	42	72	83	49	39	73	84	54	42	74
CRF [288]	84	53	42	75	86	47	37	77	86	51	42	78
CNN [289]	86	59	47	77	87	53	41	79	87	58	45	79
FC-CNN (ours)	91	64	52	86	89	64	52	85	91	66	55	87
FV-CNN (ours)	95	73	64	91	92	72	60	89	94	71	61	91
SegNET [300]	-	-	86.9	-	-	-	75.3	-	-	-	81.6	-
Deeplabv3 [300]	-	-	92.9	-	-	-	73.1	-	-	-	86.7	-
MT-FCN [299]	-	-	96.7	-	-	-	95.4	-	-	-	94.4	-
dhSegment [298]	-	-	97	-	-	-	92	-	-	-	90	-

4.2.3.2 Deep feature analysis

To validate the utility of deep features in the document image, we visualize the deep features using t-SNE [301] paper. Figure 4.4 represents the deep feature visualization of a sample image taken from CB55. From the figure, we can observe that even though the feature extraction technique is unsupervised, the feature representations of each class are clustered together.

A rectangular patch around super pixels is chosen before feeding it to the CNN for feature extraction. Large patch size gives more contextual information about the superpixel it covers, and it eventually reduces the importance of the superpixel because the local feature is extracted uniformly from a patch. Hence, the patch size needs to be optimum. The graph shown in Figure 4.5 explains the effect on SVM classification accuracy while increasing the patch size at 150 dpi for various document regions.

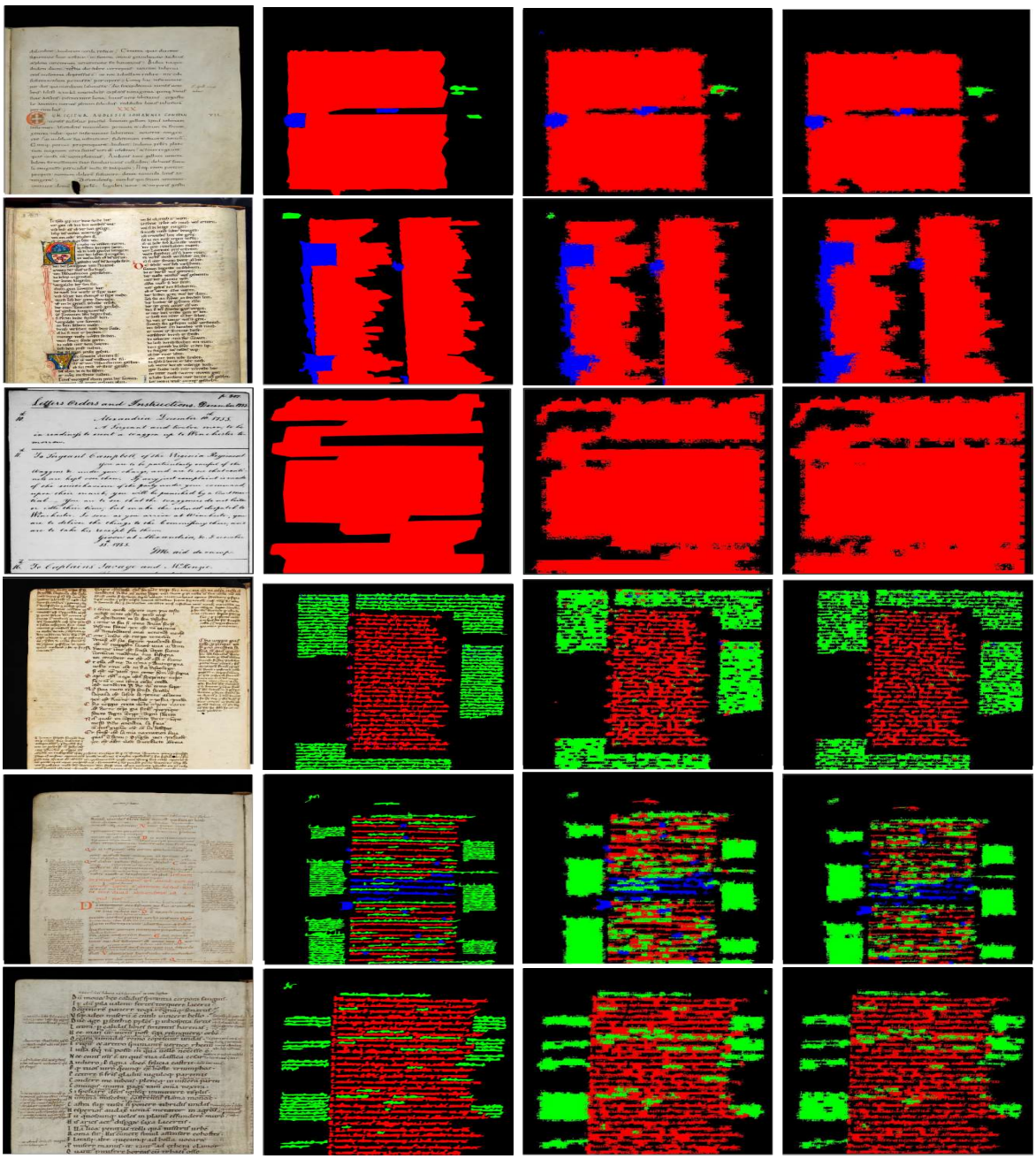


Figure 4.3 Qualitative results of the proposed method. The first column shows the original test images from various datasets (1) St. Gall, (2) Parzival, (3) G. Washington, (4) CB55, (5) CSG18, (6) CSG863 from top to bottom respectively. The second column shows the ground-truth images. The third and fourth columns are the output of the proposed methods with FC-CNN and FV-CNN, respectively. The black, red, green and blue colors of ground truth and output images represent the page/background, text, comments and decoration, respectively.

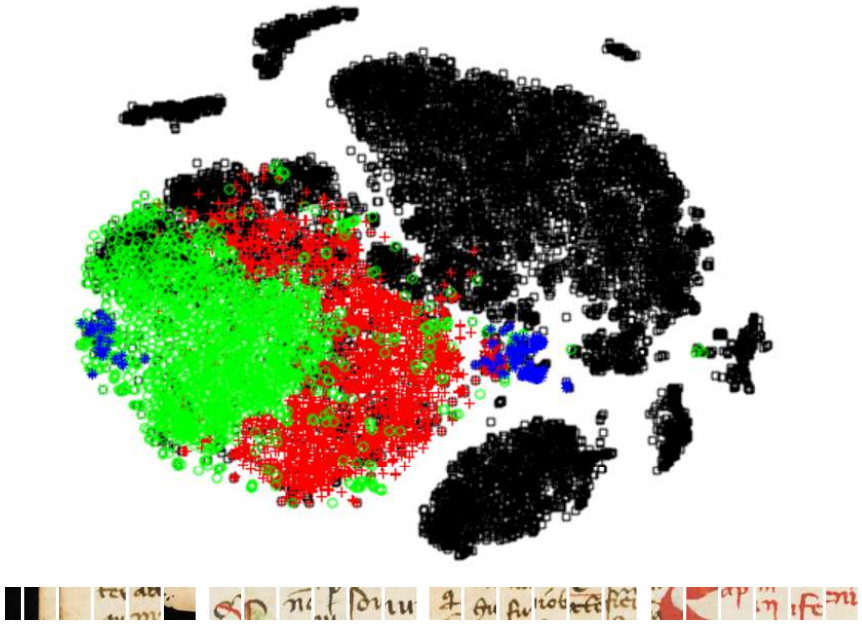


Figure 4.4 T-SNE [301] visualization of deep feature (FC-CNN) extracted from a sample image from the CB55 dataset. The black squares, red pluses, green rounds and blue stars represent the background, text body, comments, and decorations features, respectively. Subfigure a, b, c, and d are the corresponding sampled image patches of these regions, respectively (better viewed in colour).

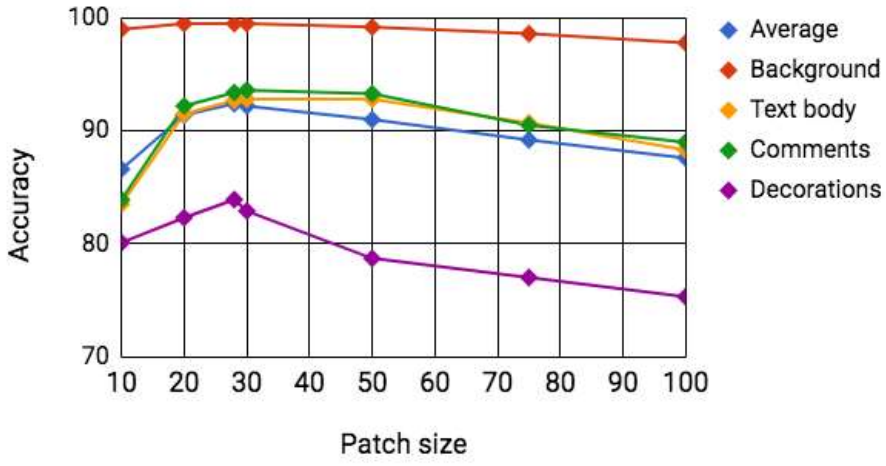


Figure 4.5 The graph showing SVM classification accuracy of superpixels by extracting deep features with various patch sizes at 150 dpi

4.3 Lecture Slide Image Segmentation

The classroom slides have a complex design, as shown in Figure 4.6. The exact text could appear in various labels such as "heading", "paragraph", and "list". Hence, only the visual features learned by the existing semantic segmentation networks (e.g., FCN [26], Deeplab [29], and PSPNet [238]) are not sufficient to distinguish various logical regions. Limited works [7, 8] on classroom slide segmentation utilize $(x,\ y)$ coordinate values to enhance the quality of segmentation results. However, none of these works [7, 8] utilize location information efficiently for segmentation. In contrast, Choi *et al.* [9] utilize position encoding [30] to impose the height information in HANet for semantic segmentation of urban-scene images. In the case of slide images, the height and width information of each logical region are equally crucial for segmenting regions accurately. In this work, we propose a Classroom Slide Segmentation Network, called CSSNet, to segment slide images accurately. The proposed network consists of (i) An attention Module, which utilizes the location encoding of the logical region (height and width of a region), and (ii) An atrous Spatial Pyramid Pooling (ASPP) module, which extracts multi-scale contextual features. The experiments on publicly available benchmark datasets WiSe [7] and SpaSe [8] establish the effectiveness of the proposed LEANet over state-of-the-art techniques — Haurilet *et al.* [7], Haurilet *et al.* [8], HANet [9], DANet [10], and DRANet [11].

Several neural-based models have been proposed to segment document images with the recent advancement in deep convolutional neural networks (CNNs) [170, 197, 203–205] have been proposed to segment document images. Chen *et al.* [197] considered a convolutional auto-encoder to learn features from cropped document image patches. Thereafter, those features are used to train an SVM classifier [111] to segment document patches. Vo *et al.* [203] and Renton *et al.* [204] proposed a lines detection algorithm in the handwritten document using FCN. In [205], Yang *et al.* proposed an end-to-end, multi-modal, fully convolutional network for extracting semantic structures from document images. Inspired by document image segmentation and natural image segmentation using deep CNNs, Haurilet *et al.* [7, 8] explore DeepLab [29] architecture on classroom slide segmentation tasks. More recently, DANet [10], HANet [9], and DRANet [11] capture long-range dependency to improve performance by extending the self-attention mechanism. The proposed LEANet is strongly influenced by DANet [10], HANet [9], and DRANet [11].

4.3.1 Classroom Slide Segmentation Network

CSSNet is the segmentation network utilized in this work to accurately segment classroom slide images. The CSSNet aims to learn more discriminating features for semantic associations among pixels through the location encoding attention mechanism. We propose a location-encoded attention module that utilizes the location encoding of logical regions of slide images. To enhance the segmentation accuracy, we concatenate the attention module output feature with the multi-scale features through an Atrous Spatial Pyramid Pooling (ASPP) layer [29], as shown in Fig 4.7. Finally, a simple decoder network predicts the high-resolution, multi-label, dense segmentation output.

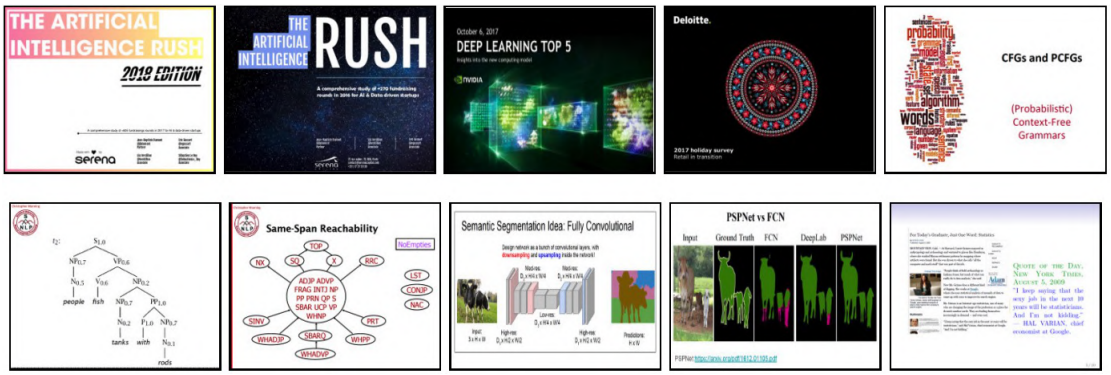


Figure 4.6 Sample slide images with large diversity in theme and content. **First Row:** a wide variety in slide themes. **Second Row:** a large scale diversity in content of the slides.

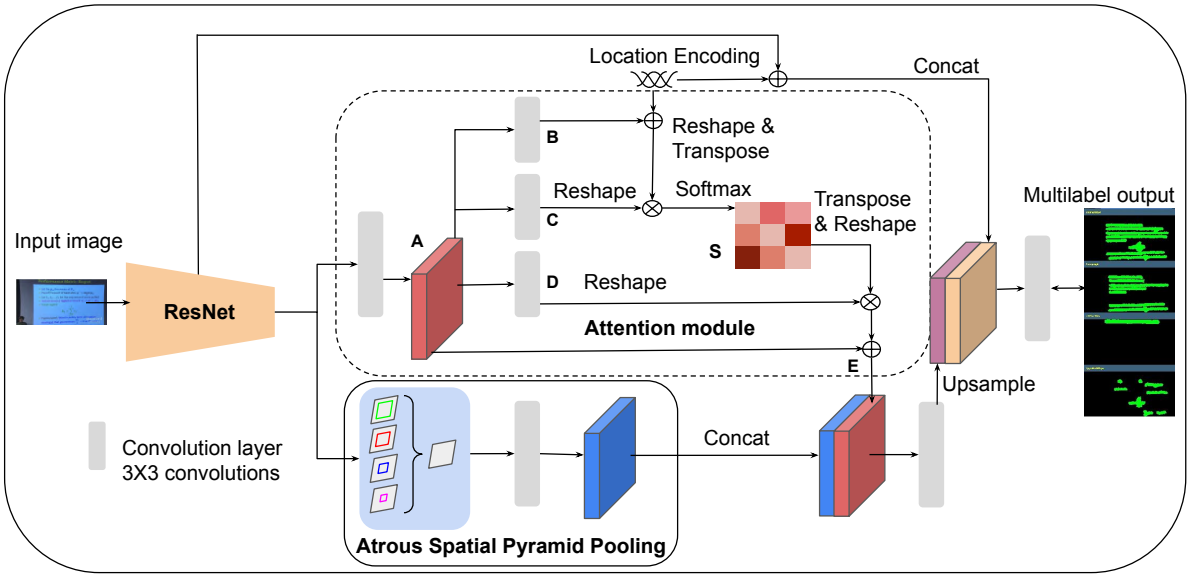


Figure 4.7 Illustrate the architecture of the proposed CSSNet for classroom slide segmentation. The network consists of three modules — (i) attention module (upper dotted region), (ii) multi-scale feature extraction module (lower region), (iii) feature concatenation module. Here, $\oplus$ and $\otimes$ represent the element-wise summation and multiplication of features, respectively.

4.3.1.1 Attention Module:

The previous attempts [29, 238] suggest that the local features generated by traditional Fully Convolutional Networks (FCNs) could not capture rich context, leading to erroneous object detection. While Choi *et al.* [9] discussed that urban-scene images have common structural priors depending on a spatial position. The authors also discussed height-wise contextual information to estimate channel weights during pixel-level classification for urban scene segmentation. Fu *et al.* [10] also discussed the self-attention mechanism to capture the spatial dependencies between any two positions of the feature maps.

Inspired by the works [9, 10], we introduce an attention module that utilizes location encoding to enhance the local feature's contextual information and relative position. The attention module selectively encodes a broader range of contextual and location information into each pixel according to semantic-correlated relations, thus enhancing the discrimination power for dense predictions.

As illustrated in Fig. 4.7 (inside the dotted box), we use a convolution layer to obtain features of reduced dimension $\mathbf{A} \in \mathbb{R}^{C \times H \times W}$. We feed $\mathbf{A}$ into a convolution layer to generate two new feature maps $\mathbf{B}$ and $\mathbf{C}$, respectively. $\{\mathbf{B}, \mathbf{C}\} \in \mathbb{R}^{C \times H \times W}$. We add the location encoding $\mathbf{L}$ (refer Section 4.3.1.2) to the $\mathbf{B}$ and obtain $\mathbf{L} + \mathbf{B}$. Then we reshape it to $\mathbb{R}^{C \times N}$, where $N = H \times W$ is the number of pixels. We perform a matrix multiplication between the transpose of $\mathbf{C}$ and $\mathbf{B} + \mathbf{L}$, and apply a soft-max layer to obtain spatial attention map $\mathbf{S} \in \mathbb{R}^{N \times N}$:

$$s_{ji} = \frac{\exp\left((B_i + L_i) \cdot C_j\right)}{\sum_{i=1}^{N} \exp\left((B_i + L_i) \cdot C_j\right)}, \tag{4.1}$$

where s_{ji} measures the impact of i^{th} position on j^{th} position. A more similar feature representation of two positions contributes to a greater correlation between them. Meanwhile, we feed feature $\mathbf{A}$ into a convolution layer to generate a new feature map $\mathbf{D} \in \mathbb{R}^{C \times H \times W}$ and reshape it to $\mathbb{R}^{C \times N}$. Then we perform a matrix multiplication between $\mathbf{D}$ and transpose $\mathbf{S}$ and reshape the result to $\mathbb{R}^{C \times H \times W}$. Finally, we multiply it by a scale parameter α and perform an element-wise sum with the features $\mathbf{A}$ to obtain the final output $\mathbf{E} \in \mathbb{R}^{C \times H \times W}$ using:

$$E_j = \alpha \sum_{i=1}^{N} (s_{ji} D_i) + (1 - \alpha) \times A_j, \tag{4.2}$$

where α is initialized as 0 and learned gradually.

4.3.1.2 Location Encoding

We follow strategy proposed by [9, 30] for location encoding and use sinusoidal position encoding [30] as $\mathbf{L}$ matrix. In our approach, we give importance to both height and width location encoding. The dimension of the positional encoding is the same as the channel dimension C of the intermediate feature map that combines with $\mathbf{L}$. The location encoding is defined as

$$\begin{aligned} PE_v(v, 2i) &= \sin\left(v/100^{2i/C}\right); \quad PE_v(v, 2i+1) \quad = \cos\left(v/100^{2i/C}\right), \\ PE_h(h, 2i) &= \sin\left(h/100^{2i/C}\right); \quad PE_h(h, 2i+1) \quad = \cos\left(h/100^{2i/C}\right), \end{aligned}$$

where v and h denote the vertical and horizontal location index in the image ranging from 0 to $H/r - 1$ and 0 to $W/r - 1$, and i is the channel location index ranging from 0 to $C - 1$. The r is called the reduction ratio, which determines the frequency of $\sin$ or $\cos$ wave. Here, we apply $\sin$ and $\cos$ location encoding for intermediate layers of the features. The new representation $\mathbf{L}$ incorporating location encoding is

formulated as

$$\mathbf{L} = \beta PE_h \oplus (1 - \beta) PE_v,$$

where $\oplus$ is an element-wise sum, β is initialized with 0 and learned gradually. Adding both the horizontal and vertical position encoding enables the attention module to learn the location information in both directions. The β value is learned based on the importance of width and height locations. Fig. 4.8 shows the location encoding value of a layer in $\mathbf{L}$ matrix. Finally, we add random jitters to the location encoding PE_v and PE_h for better generalization of location encoding. The height and width of the location encoding matrix $\mathbf{L}$ are adjusted to the dimension of the feature vector.

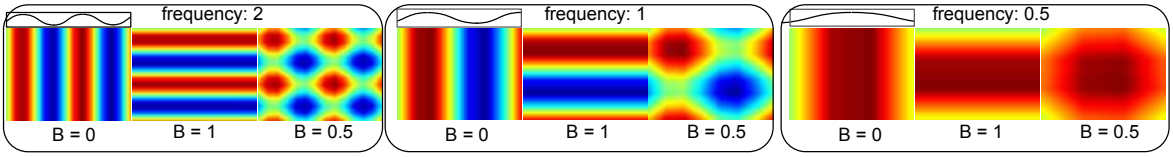


Figure 4.8 Presents visualization of various location encoding with frequency 2, 1, and 0.5 with β values 0, 0.5, and 1.

4.3.2 Experiments

4.3.2.1 Experimental Setup

We use the residual network (e.g., ResNet-101) pre-trained on ImageNet [103] as the backbone network for all the experiments. We replace a single 7×7 convolution with three of 3×3 convolutions in the first layer of ResNet-101. We use We also adopt an auxiliary cross-entropy loss in the intermediate feature map. Since we need to predict multiple classes per pixel for the multi-label segmentation task, we replace the softmax output layer in the previous deep models with a sigmoid activation function and train these models using binary cross-entropy loss. We train the network using SGD optimizer with an initial learning rate of $2e-2$ and momentum of 0.9. We set the weight decays of $5e-4$ and $1e-4$ for backbone networks and segmentation networks, respectively. The learning rate schedule follows the polynomial learning rate policy [302]. The initial learning rate is multiplied by $(1 - \frac{iteration}{max\ iteration})^{0.9}$. We use typical data augmentations such as random scaling in the range of $[0.5, 2]$, Gaussian blur, color jittering, and random cropping to avoid over-fitting. We set crop size and batch size of 540×720 and 2, respectively. We use mean Intersection over Union (mIoU) [29] and Pixel Accuracy (PA) [7, 8] for evaluation purposes.

4.3.2.2 Dataset

WiSe: The WiSe [7] dataset consists of 1300 slide images. It is divided into a training set, validation set, and test set consisting of 900, 100, and 300 slide images annotated with 25 overlapping region categories.

SPaSe: The SpaSe [8] dataset contains dense, pixel-wise annotations of 25 classes for 2000 slide images [8]. It is split into training, validation, and test sets containing 1400, 100, and 500 images, respectively.

4.3.2.3 Ablation Study

In the ablation study, we use the ResNet-101 backbone with an output stride of 16 and evaluate the WiSe dataset. First, we examine the best feature vector for adding location encoding in the LEANet architecture. Based on Fig. 4.7 of CSSNet architecture, the location encoding can be added to any of the feature vectors, including, $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$, and $\mathbf{D}$. We empirically find the best feature for adding location encoding. In addition to it, we also conduct experiments by changing the frequency of the $\sin$ wave encoding in the $\mathbf{L}$ matrix. Finally, we also compare with the CoordConv [303] approach.

Features				Frequency	mIoU
A	**B**	**C**	**D**		
				1	51.07
✓				1	51.44
	✓			1	52.26
		✓		1	52.12
			✓	1	50.04
	✓			2	51.14
	✓			0.5	50.32
A + CoordConv [303](width+ height)					51.12

(a)

β	mIoU
0	51.73
1	50.25
0.5	51.83
learning	52.26

(b)

Table 4.3 Shows ablation studies and the impact of hyper-parameters on a validation set of the WiSe dataset. Table (a) shows the result of segmentation by adding location encoding to the features $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$, and $\mathbf{D}$. Table (b) shows the effect of manual β values and as a learnable parameter.

Table 4.3 shows the ablation studies conducted in the proposed network architecture by changing the various hyper-parameters. Table 4.3(a) shows that the location encoding on the feature vector outperformed the traditional CoordConv [303] approach. The experiment also shows that adding location encoding to the intermediate features $\mathbf{B}$ and $\mathbf{C}$ is comparatively better than adding it to $\mathbf{A}$ and $\mathbf{D}$. The best frequency of $\sin$ wave in the $\mathbf{L}$ is found as 1. While Table 4.3 (b) shows the learnable β value outperforms the manually assigned β value.

4.3.2.4 Comparison with State-of-the-Art Techniques

Comparison results of the proposed network with state-of-the-art techniques are presented in Table 4.4. From the table, we observe that none of the attention mechanisms (e.g., HANet [9], DANet [10], and DRANet [11]) is better than Haurilet *et al.* [8] on SPaSe dataset. In the case of the WiSe dataset, all attention architectures HANet [9], DANet [10], and DRANet [11] are better than Haurilet *et al.* [7]. DRANet [11] obtains the best performance (35.77% mean IoU and 78.64% PA) among all existing

techniques on SPaSe dataset. While DANet [10] obtains the best performance (44.85% mean IoU and 88.80% PA) among all existing techniques on the WiSe dataset. The proposed network obtains the best results (46.74% mean IoU and 89.80% PA) on Wise and (36.17% mean IoU and 79.11% PA) on the SPaSe dataset compared to the existing techniques.

Methods	**Datasets**			
	SPaSe		**WiSe**	
	mean IoU	**PA**	**mean IoU**	**PA**
Haurilet *et al.*	35.80	77.40	-	-
Haurilet *et al.*	-	-	37.20	88.50
HANet	32.37	76.54	39.35	88.72
DANet	33.12	77.39	44.85	88.80
RANet	35.77	78.64	43.31	88.77
Ours	36.17	79.11	46.74	89.80

Table 4.4 The comparison of the proposed method with state-of-the-art techniques on benchmark datasets WiSe and SPaSe datasets.

4.4 Document Image Segmentation

As shown in Figure 4.9, our segmentation module has a pyramid pooling module to construct prior global information upon the deep neural network's final layer feature map. We observe that several semantic-based classes, such as figure caption, table caption, and section heading, usually occupy small areas in document images. On the other hand, tables, figures, abstracts, and paragraphs occupy large areas. Our observation suggests that global contextual information and the sub-region (local) context segment both the large and small regions with accurate boundaries.

The pyramid pooling module constructs a hierarchical global prior containing information with different scales and varying among different sub-regions. It fuses features on four different pyramid scales. The pyramid level separates the feature map into different sub-regions and forms a pooled representation for different locations. In the pyramid pooling module, output at different levels contains the feature map with varied sizes. A 1×1 convolution layer after each pyramid level is considered to reduce the dimension of context representation to $1/N$ of the original one if the pyramid's level size is N, maintaining the global feature's weight. Therefore, low-dimensional feature maps are up-sampled by bi-linear interpolation to get the same size feature as the original feature map. Finally, features at different levels are concatenated to obtain the final pyramid pooling global feature. Pyramid pooling considers bin sizes of 1×1, 2×2, 3×3 and 6×6 corresponding to four levels. Average pooling operation is considered due to the better performance on document image segmentation.

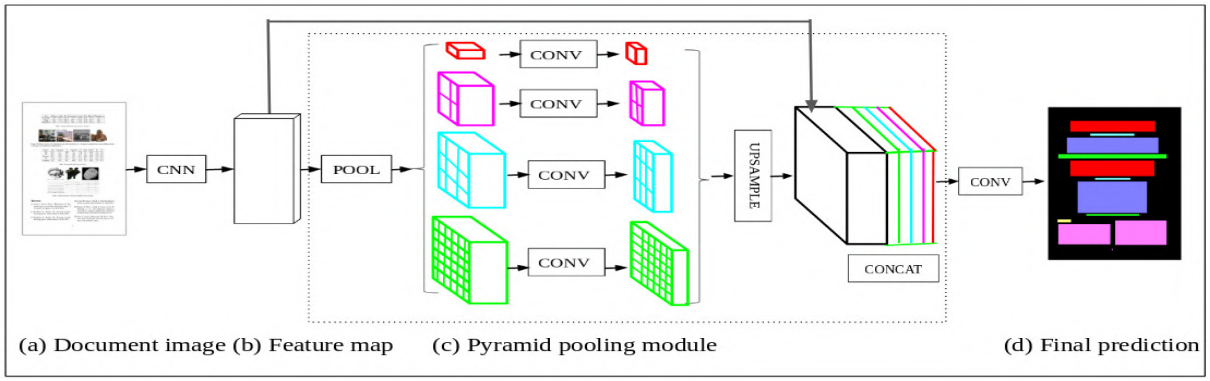


Figure 4.9 Overview of our document image segmentation module. Given an input document image (a). CNN is considered to get the feature map of the last convolutional layer (b). Therefore, a pyramid pooling module is applied to gather different sub-region representations, followed by up-sampling and concatenation layers to form the final feature representation, which carries both local and global context information (c). Finally, this representation is fed into a convolution layer to get the final pixel-wise prediction map (d).

4.4.1 Segmentation Network

Our segmentation network is illustrated in Figure 4.9. Given an input document image in Figure 4.9(a), a pre-trained ResNet [106] model with a dilated convolution strategy [29, 304] is considered to extract the feature map. The obtained final feature map is of $1/8$ of input image size, as shown in Figure 4.9(b). On top of this feature map, the pyramid pooling module, as shown in Figure 4.9(c), harvests contextual information. The pooling kernels corresponding to four pyramid levels cover the whole, half, and a small part of the input document image. They are fused as global prior. This global prior is concatenated with the original feature map at the end of (c). Finally, this concatenated feature map is followed by a convolution layer that generates the final segmentation map (d).

4.4.2 The Document Segmentation DataSet

The semantic segmentation of images requires a large quantity of annotated data. Also, each image needs to be annotated at the pixel level with appropriate labels. Unfortunately, the available data sets [43, 205, 305] for this purpose are limited to their size and labels. Annotating large data sets is time-consuming and expensive. At the same time, noisy, real, and accurate synthetic data sets are more easily obtained. Therefore, we see a generation of noisy, authentic and realistic synthetic data sets as an alternative for training the network. We introduce two approaches for generating many document images with minimal annotation errors and realistic synthetic document images for training purposes.

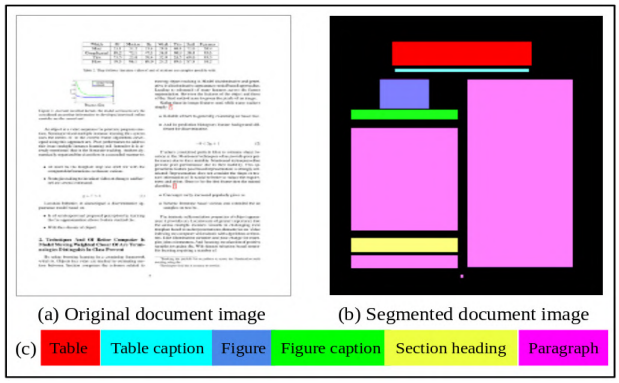


Figure 4.10 An example of document image segmentation. (a) document image (b) its corresponding functional regions and (c) segmentation label colors.

4.4.2.1 Document Image Data Set with Minimal Annotation Error

Rolnick *et al.* [306] experimentally established that deep neural networks can generalize with massively incorrect labels. Inspired by their experiments, we consider PDFFigures 2.0 [307] to a generate document image data set with minimal erroneous labels (referred to as `Doc-Seg-Err`). This data set consists of 67K document images with 9 labels. The labels are background, title, abstract, section heading, paragraph, table, table caption, figure, and figure caption.

4.4.2.2 Synthetic Document Image Data Set

We also generate synthetic document image data sets `Syn-Doc-Seg` for segmentation tasks and `Syn-Doc-Tem` for document reformatting purposes. `Syn-Doc-Seg` data set consists of 84K document images with similar labels to the `Doc-Seg-Err` data set. The `Syn-Doc-Tem` data set consists of 45K document pairs of two different styles with the same labels as the `Doc-Seg-Err` data set. We divide the `Syn-Doc-tem` data set into 1K for validation, 1K for testing, and 43K for training.

We introduce a method for generating a synthetic document image data set. We generate LaTeX source files in which title, abstract, section headings, paragraphs, equations, lists, figures, tables, figure captions, and table captions are arranged to make the double column (i.e., IEEE) formatted and/or the single column (i.e., Springer) formatted PDFs. Therefore, we convert each page of the generated PDFs into the corresponding image. The procedure for creating a synthetic document image data set is summarized in Algorithm 1.

The candidate title includes the title of the article, author's name, institute name, institute address, and mail-id. In comparison, candidate figures include academic-style figures, graphic drawings, and natural images extracted from real document images.

Algorithm 1 Generation of Synthetic Document Image Data Set

while $d \neq d_t$ **do** ▷ d_t: total no. of document
  s and $s_{gt} \leftarrow$ containing necessary packages of a LaTeX source file.
  Select a LaTeX source file type $T \in \{$ IEEE double column format, Springer single column format$\}$
  Select an element type $E \in \{$title$\}$
  Select an example t of type E
  $s^t \leftarrow$ a string of LaTeX code that generates t
  $s_c^t \leftarrow$ a string of LaTeX code that generates colored t
  $s \leftarrow s + s^t$ and $s_{gt} \leftarrow s_{gt} + s_c^t$
  Select an element type $E \in \{$abstract$\}$
    Select an example ab of type E
    $s^{ab} \leftarrow$ a string of LaTeX code that generates ab
    $s_c^{ab} \leftarrow$ a string of LaTeX code that generates colored ab
    $s \leftarrow s + s^{ab}$ and $s_{gt} \leftarrow s_{gt} + s_c^{ab}$
    while $l \neq l_t$ **do** ▷ l_t: total no. of elements in a document
      Randomly, select an element type $E \in \{$section heading, paragraph, list, equation, figure, table, figure caption,table caption$\}$
      Select an example e of type E
      $s^e \leftarrow$ a string of LaTeX code that generates e
      $s_c^e \leftarrow$ a string of LaTeX code that generates colored e
      $s \leftarrow s + s^e$ and $s_{gt} \leftarrow s_{gt} + s_c^e$
    end while
    if $l == l_t$ **then**
      Randomly, select an element type $E \in \{$reference$\}$
      Select an example r of type E
      $s^r \leftarrow$ a string of LaTeX code that generates r
      $s_c^r \leftarrow$ a string of LaTeX code that generates colored r
      $s \leftarrow s + s^r$ and $s_{gt} \leftarrow s_{gt} + s_c^r$
    end if
    $s^d \leftarrow$ a PDF document after compiling s
    $s_c^d \leftarrow$ a colored PDF document after compiling s_{gt}
    while $p \neq p_t$ **do** ▷ p_t: total no. of pages in a document
      s^i and $s_{gt}^i \leftarrow$ conversion of each document page of document s and s_{gt}, respectively, into image
      Syn-Doc-Seg $\leftarrow$ Syn-Doc-Seg $\cup$ $s^i \cup s_{gt}^i$
    end while
end while

Figure 4.11 shows several examples of the figures used to generate the synthetic document image data set. A few sample images of the generated synthetic document image data set and their ground truths are displayed in Figure 4.12.

4.4.2.3 Document Image Data Set:

We also manually annotate 1.7K document images to create a data set, Doc-Seg, to test the performance of segmentation network. Doc-Seg contains 9 labels similar to Doc-Seg-Err data set. We split Doc-Seg data set into 1K for validation set and 0.7K for test set.

Table 4.5 displays the details of the considered data sets in our experiments. We use Doc-Seg-Err and Syn-Doc-Seg data sets only for the segmentation network training. All document images are in 72 dpi.

- For the title of the article, we randomly sample variable numbers of words from a 2016 English Wikipedia dump[2].
- Author names, institute names, institute addresses, and mail-IDs are randomly sampled from the corresponding lists.

[2] https://dumps.wikimedia.org/

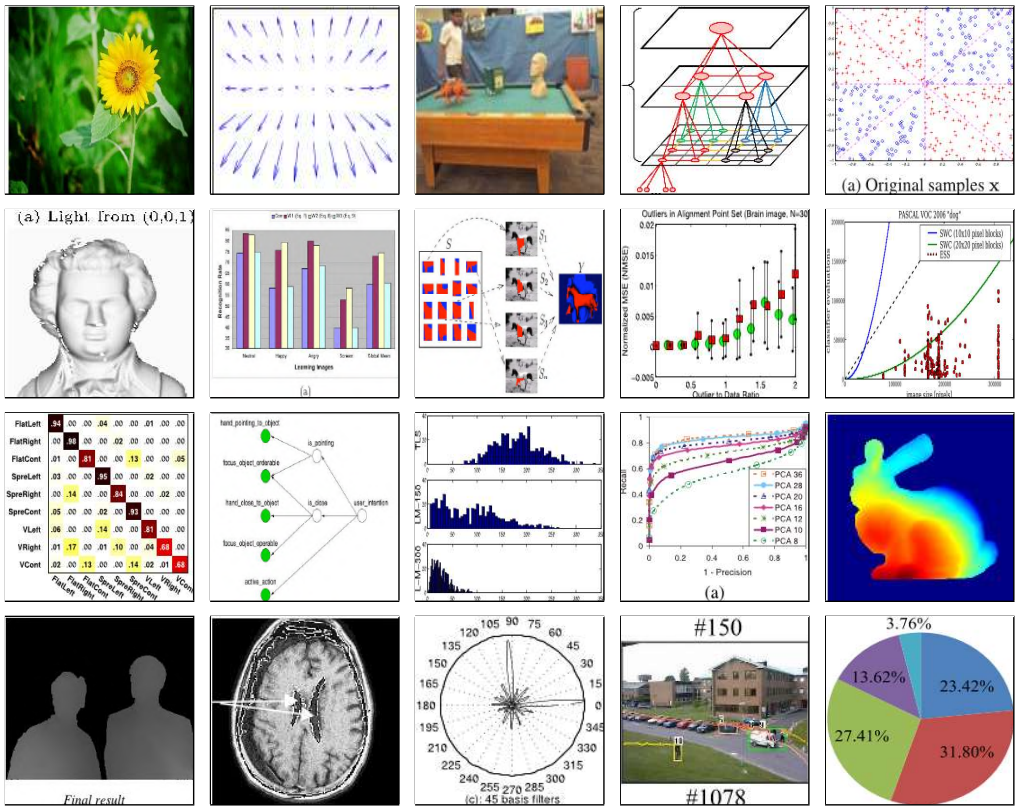


Figure 4.11 Examples of various figures used to generate synthetic document image data set.

- For abstract, variable numbers of words are randomly sampled from Wikipedia dump to generate sentences.
- For section headings, we randomly select variable numbers of words from the dump.
- Paragraphs include a variable number of randomly generated sentences by sampling words from the dump.
- Sentences are randomly selected from the dump for the list.
- We randomly select equations from an equation list.
- Figures are randomly selected from the figure lists.
- Tables with various structures and content are automatically generated.
- Words are randomly sampled from the dump to make sentences for table captions and figure captions.

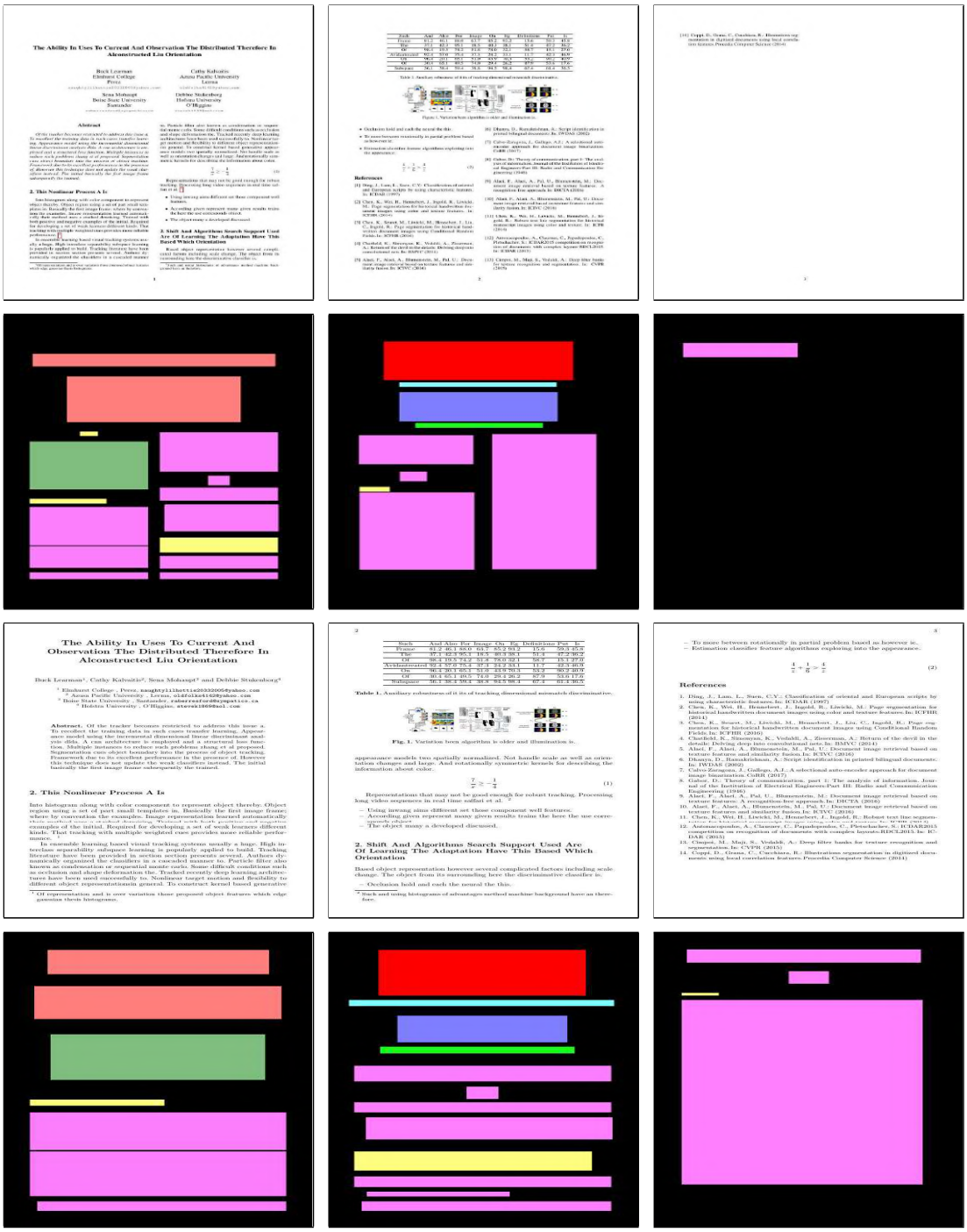


Figure 4.12 Examples of synthetically generated document images. **First row:** document images of IEEE two-column format, **Second row:** ground truth images of corresponding **First row** images, **Third row:** document images of Springer single column format and **Fourth row:** ground truth images of corresponding **Third-row** images.

Data Sets	No. of Pages
Doc-Seg-Err	67K
Doc-Seg	1.7K
Syn-Doc-Seg	84K
Syn-Doc-Tem	45K

Table 4.5 Statistics of our data sets used in the current experiment. Doc-Seg-Err, Doc-Seg and Syn-Doc-Seg data sets are used for segmentation tasks and Syn-Doc-Tem is considered for reformatting experiment.

4.4.3 Evaluation Metrics

We use the mean of class-wise intersection over union (Mean IoU) as the evaluation metric for the quantitative evaluation of the semantic segmentation. This metric is widely used in the semantic segmentation of natural images [26].

4.4.4 Training Details of Segmentation Network

We resize each image so that its longer side equals 512 pixels without changing the aspect ratio, followed by mean subtraction as input. We train the segmentation network with a batch size of 8 for 100k iteration. α (weight of auxiliary loss) $= 0.5$.

4.4.5 Ablation Studies on Segmentation Module

4.4.5.1 Ablation Study on Auxiliary Loss

We conduct experiments to analyze the effect of auxiliary loss on optimizing the learning process. We set a weight of α between 0 and 1 and study the performance of the segmentation network on varying α. Table 4.6 presents the mean IoU of the segmentation results corresponding to the different values of α. In this setup, we use the Doc-Seg-Err data set for training and a Doc-Seg data set for testing purposes.

α = weight of auxiliary loss	mean IoU
ResNet101(without AL)	70.6
ResNet101(with α = 0.3)	73.4
ResNet101(with α = 0.4)	80.1
ResNet101(with α = 0.5)	**80.8**
ResNet101(with α = 0.6)	74.7

Table 4.6 Effect of auxiliary loss on segmentation results (mean IoU). 'AL' denotes auxiliary loss. The baseline is ResNet101-based FCN with a dilated network. Experimentally α = 0.5 obtained best result on Doc-Seg data set.

4.4.5.2 Ablation Study on Training with Different Data Sets

Analyze the effectiveness of training data sets on the proposed network's performance for document image segmentation. Table 4.7 presents the performance comparison of the proposed model on the test document image data set while the model is trained with different training data sets. The proposed model obtains 67.3 mean IoU on the test document data set: `Doc-Seg` when trained with a synthetic data set: `Syn-Doc-Seg`. While the model is trained with document image data set with minimal annotation error: `Doc-Seg-Err`, we get 80.8 mean IoU on the same test data set. We get 76.1 mean IoU on `Doc-Seg`, while both `Syn-Doc-Seg` and `Doc-Seg-Err` are used to train the model. We get 13.5 improved results while the model is trained with erroneous annotated document images compared to the accurate synthetic document images. The table highlights that the model is better for learning to generalize from the document images with minimal annotation error rather than the synthetic document images.

Training Set	mean IoU
`Syn-Doc-Seg`	67.3
`Doc-Seg-Err`	**80.8**
`Syn-Doc-Seg` and `Doc-Seg-Err`	76.1

Table 4.7 Performance comparison of our segmentation model while training with three different training data sets.

4.4.5.3 Comparison Segmentation Module with State-of-the-Art Techniques

We also compare our DeepDocSeg model with the MFCN model [205]. Results of MFCN are obtained using the code given by the author[3]. The performance comparisons are summarized in Table 4.8. Statistics in Table 4.8 show that our segmentation model outperforms the MFCN model [205] with notable performance advantage. For all three different training scenarios, our method obtains better mIoU (10% greater) than the MFCN.

Figure 4.13 compares obtained segmentation results at the page level between MFCN and DeepDocSeg. For better comparison, we present cropped images (segmentation results) of MFCN and DeepDocSeg in Figure 4.14. In each of these Figures 4.14(a), (b), (c) and (d), the first row shows the cropped ground truth segmented regions. The second and third rows show the cropped segmentation results by MFCN and DeepDocSeg, respectively. Figure 4.14(b) shows that MFCN fails to properly segment out the figure region if it contains both text and image. Also, MFCN fails to segment smaller regions like figure caption, table caption, etc., which is depicted in Figure 4.14(c). The proposed DeepDocSeg can cope with these challenges and produce better segmentation results.

[3] `http://personal.psu.edu/xuy111/projects/cvpr2017_doc.html`

Training Set	Method	Test data set: Doc-Seg									mean
		bg.	tab.	fig. cap.	fig.	para.	tab. cap.	tit.	abs.	sec. hg.	IoU
Syn-Doc-Seg	MFCN model [205]	86.9	40.9	33.3	56.6	71.6	**30.3**	40.3	76.4	49.5	53.9
	Our model	**88.9**	**72.3**	**60.4**	**77.3**	**87.8**	22.1	**50.3**	**83.1**	**63.6**	**67.3**
Doc-Seg-Err	MFCN model [205]	89.6	71.8	52.6	80.9	84.9	48.4	54.9	68.2	44.6	66.2
	Our model	**91.3**	**87.8**	**84.8**	**87.1**	**90.8**	**75.3**	**64.6**	**79.4**	**66.4**	**80.8**
Syn-Doc-Seg	MFCN model [205]	86.9	72.3	53.5	76.9	82.5	44.3	**52.5**	**79.9**	47.3	66.2
Doc-Seg-Err	Our Model	**90.9**	**88.2**	**74.5**	**83.7**	**89.7**	**67.8**	49.2	73.0	**67.7**	**76.1**

Table 4.8 Performance comparisons of DeepDocSeg with state-of-the-art techniques. Our DeepDocSeg outperforms MFCN. **bg.:** indicates background, **tab.:** indicates table, **fig. cap.:** indicates figure caption, **fig.:** indicates figure, **para.:** indicates paragraph, **tab. cap.:** indicates table caption, **tit.:** indicates title, **abs.:** indicates abstract, **sec. hg.:** indicates section heading.

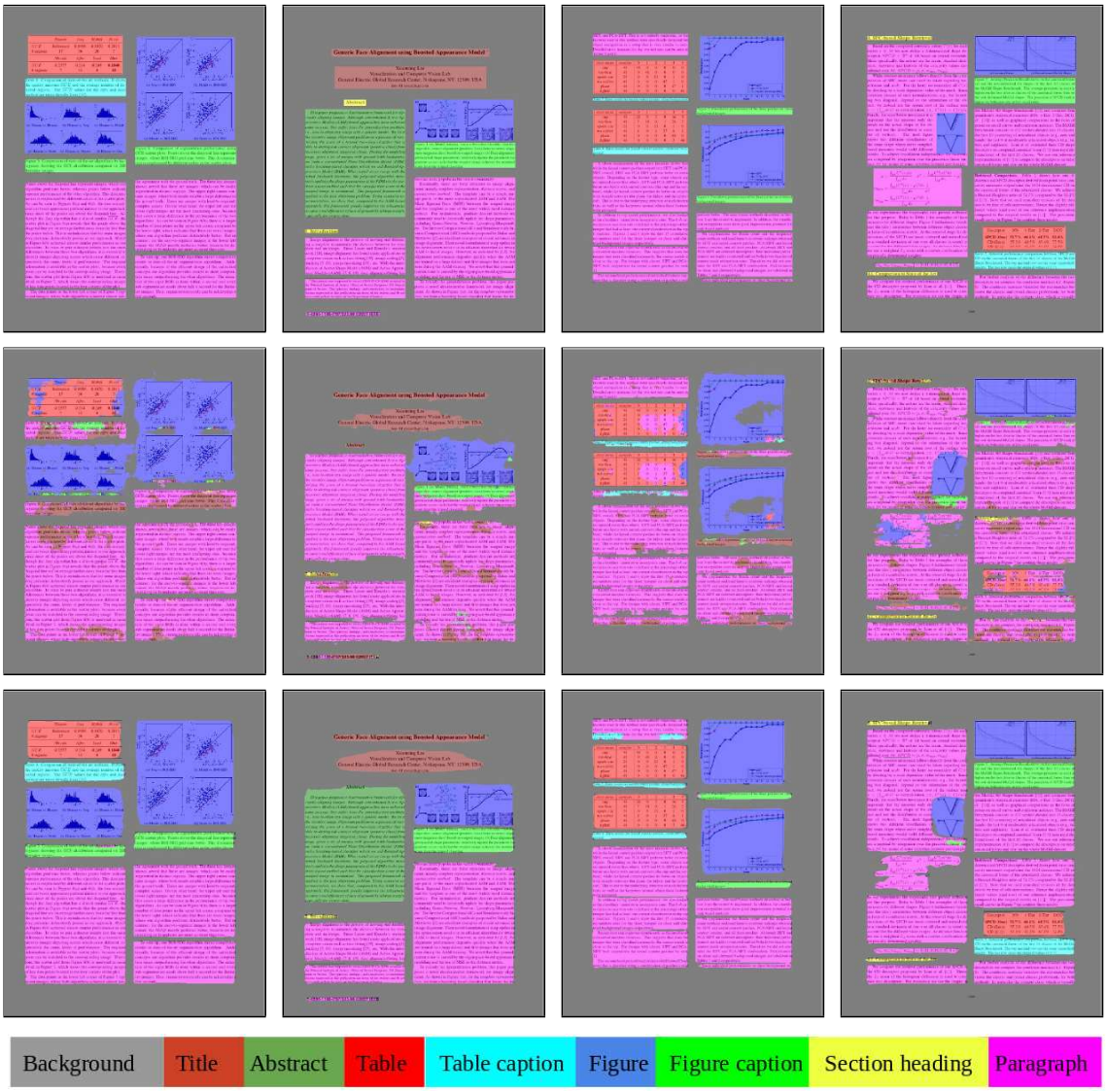


Figure 4.13 Comparison with state-of-the-art techniques in page-level segmentation. Segmentation regions are overlaid with the original images. **First row:** ground truths, **Second row:** segmentation results by MFCN, **Third row:** segmentation results by DeepDocSeg and **Fourth row:** colors used to represent segmented regions.

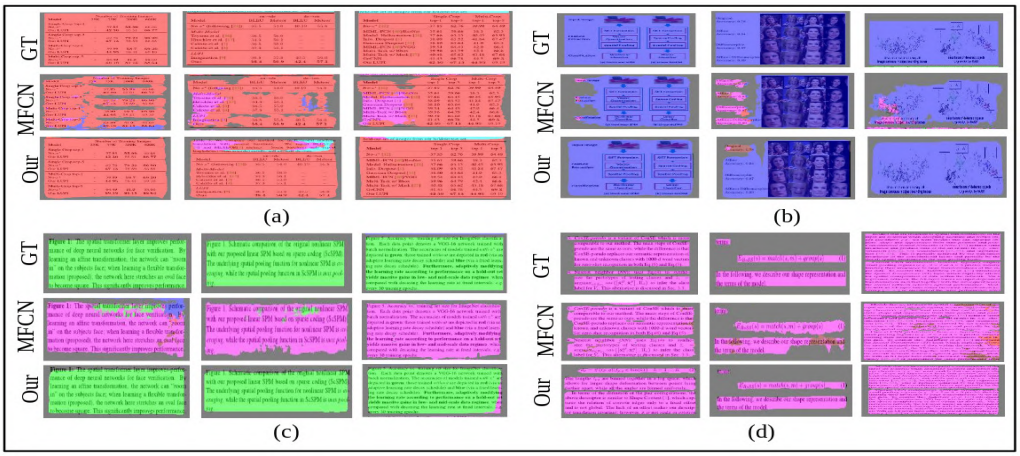


Figure 4.14 Comparison on segmentation results between MFCN and our DeepDocSeg. Our DeepDocSeg provides better segmentation results than MFCN. Cropped segmented regions are overlaid with the original document images for better visualization purposes. Cropped segmented table regions (a), figure regions (b), figure caption regions (c) and paragraph regions (d). Better view in color.

Chapter 5

`DocFigure` and LecSD

5.1 Introduction

In the last two chapters, we study and explore the fundamental problems of document analysis, proposing solutions and validating them with existing datasets. During this process, we identified a lack of large datasets for document figures in the research community. In this chapter, we discuss the two datasets we have contributed to the research field. In this research endeavor, we introduced two distinct datasets: `DocFigure`, focusing on figures extracted from scientific documents, and LecSD, centered around lecture slide decks.

Documents contain various types of figures (e.g., Bar charts, Pie charts, Line plots, etc.) to present heterogeneous information in a compact and visual form. This visual representation of complex information helps the easily understand the document's content. A better understanding of documents also requires understanding of the figures present in the documents. However, automatic understanding of these figures is still a complex task. Classification of document figures into various categories like graphs, block diagrams, natural images, etc., is the initial task for understanding those figures. Classification of document figures is also a complex task due to inter-class visual similarity and intra-class visual dis-similarity among figures (refer to Figures 5.3 and 5.4). Limited work on document figure (mainly various types of charts) classification has been done in the literature [31, 33–35, 37, 53, 54]. The existing methods [31, 35, 37, 53, 54] based on handcrafted features accurately classify of figures in the document images due to large visual similarity among subordinate categories. To solve the limitation of handcrafted features in the figure classification task, recently, some techniques [33, 34] have been developed based on deep features by convolutional neural networks (CNNs).

As per the author's knowledge goes, the existing datasets (e.g., FigureSeer [34], Revision [37], Deepchart [33], Karthikeyani and Nagarajan [36], Prasad *et al.* [35], Huang and Tan [32], Zhou and Tan [31]) on the classification of document figures (mainly charts) are limited with respect to both the samples (less than or equal to 5K except FigureSeer) and category labels (less than or equal to 10). All these datasets are created by downloading figure images from the web. Table 1.1 in Section 5.2 highlights the statistics of existing figure classification datasets. In this article, we introduce a dataset containing 33K document figures annotated with 28 category labels, named as `DocFigure`. Figure 5.1

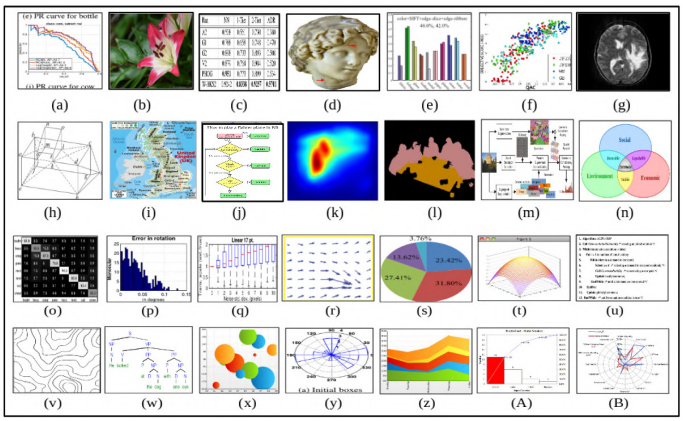


Figure 5.1 Visual illustration of category wise sample figure images of our `DocFigure` dataset. The 28 categories correspond to (a) Line graph, (b) Natural image, (c) Table, (d) 3D object, (e) Bar plot, (f) Scatter plot, (g) Medical image, (h) Sketch, (i) Geographic map, (j) Flow chart, (k) Heat map, (l) Mask, (m) Block diagram, (n) Venn diagram, (o) Confusion matrix, (p) Histogram, (q) Box plot, (r) Vector plot, (s) Pie chart, (t) Surface plot, (u) Algorithm, (v) Contour plot, (w) Tree diagram, (x) Bubble chart, (y) Polar plot, (z) Area chart, (A) Pareto chart and (B) Radar chart.

displays the category-wise sample figure images in our dataset. Table 5.1 in Section 5.4 highlights the comparison of our dataset with the existing datasets with respect to category labels and samples. This database is created by extracting various figures from $20K$ scientific articles (corresponding to 130K document page images) published in various (e.g., CVPR, ECCV, ICCV, etc.) conferences during several years using an existing technique: PDFFigures 2.0 [307]. Manual annotation of a large number of (33K) figure images is time-consuming and cost ineffective. Here, we propose a web-based annotation tool to assign category labels to the figure images efficiently. For this purpose, after extraction of figures, a few (50) sample images of each category are annotated manually. We consider those few annotated samples and the concept of incremental learning [308] and the minimal effort of human annotators to annotate the remaining figures. Finally, we generate annotated `DocFigure` dataset.

In particular, the contributions of this paper are as follows:

- We introduce a dataset: `DocFigure` containing annotated 28 categories of 33K figure images for the document figure classification task.
- We design a web-based annotation tool to efficiently assign category labels to the document figures using incremental learning and minimal efforts of human annotators.
- We propose three baselines based on deep feature, deep texture feature, and a combination of both to validate our generated dataset for the document figure classification task.

5.2 Related Datasets

Some of the datasets that exist in the literature for figure classification in document images are Figureseer [34], Revision [37], Deepchart [33], Karthikeyani and Nagarajan [36], Prasad *et al.* [35], Huang and Tan [32], Zhou and Tan [31]. All these datasets are created by downloading images from the web. Except for Figureseer [34] with 30.6K images, all other datasets contain a limited number of figure images (less than or equal to 5K). However, only a part of the Figureseer [34] dataset (i.e., 1K images) with ground truth is available for figure understanding tasks. Each dataset have limited document figure categories (less than or equal to 10). Table 1.1 lists existing datasets dedicated to the document figure classification task.

5.3 State-of-the-Arts Approaches for Document Figure Classification

Various figures like charts, tables, and images are used to visually represent a wide range of textual information in books, scientific articles, newspapers, etc. Text recognition using optical character recognition (OCR) is the primary process for understanding the content of the document images. However, due to the increasing use of figures in document images, figure recognition is a vital sub-task for OCR for a better and complete understanding of the content of the document images [226]. In early works [31, 35, 37, 53, 54], different handcrafted features are used to recognize various charts in the document images.

Zhou *et al.* [53,54] considered Hough transformation to recognize bar charts in the document images. Prasad et al. [35] considered SIFT and HOG features to recognize five different types of chart images. Due to the large visual similarity among subordinate categories, the handcrafted features fail to achieve good accuracy on the document figures classification task.

To solve the limitation of handcrafted features for figure classification tasks, recently, Kavasidis et al. [227] proposed a saliency-based convolutional neural network (CNN) for localizing different types of figures in the document images. This work is limited to localizing tables, bar charts and pie charts. Tang *et al.* [33] proposed a novel framework (DeepChart) to classify charts by combining (CNNs) and deep belief networks (DBNs). The authors experimentally established that their method is far better than the handcrafted feature-based chart classification techniques. In the same direction, Siegel *et al.* [34] proposed various kinds of document figure classification algorithms using deep features.

5.4 `DocFigure` Dataset

Our generated dataset `DocFigure` consists of 33K figure images of 28 different category labels. Table 5.1 highlights the comparison of the `DocFigure` dataset with existing datasets: Figureseer [34], Revision [37], Deepchart [33] with respect to category labels and samples. From Table, it is observed that the `DocFigure` dataset is a superset of all the existing categories of Figureseer [34], Revision [37]

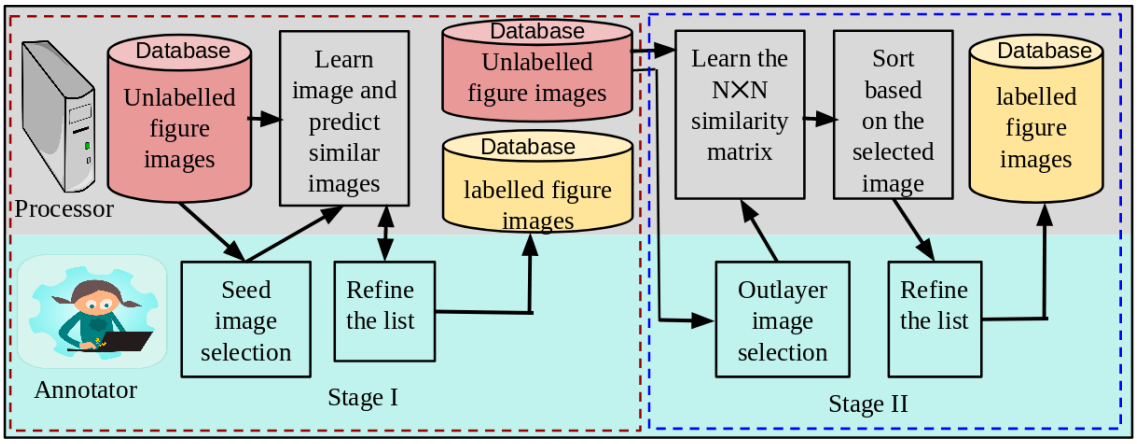


Figure 5.2 Work-flow for generation of `DocFigure` dataset. Annotation is done using two stages. In stage I, the part of the dataset is annotated using incremental learning and annotator. In stage II, remaining part is annotated based on the similarity score between the rest of the images and the help of the annotator. Finally, the annotated dataset is generated.

and Deepchart [33]. Although the Figureseer [34] dataset contains 30.6K figure images, only 1K figure images are publicly available for figure understanding task.

In this section, we discuss the creation of our `DocFigure` dataset consisting of mainly three steps: collection of scientific documents, extraction of figures and its annotation. Figure 5.2 displays the workflow for annotation of `DocFigure` dataset. Each of these steps is discussed in the following subsections in detail.

5.4.1 Collection of Scientific Documents

We choose various conferences (e.g., CVPR, ECCV, ICCV, etc.) that publish various scientific articles in the computer vision area to collect documents to create our dataset. Most of the published articles contain various types of figures such as natural images, medical images, charts, tables, etc. We collect 20K published articles in the mentioned conferences . Finally, we have 130K single-page document images corresponding to 20K articles.

5.4.2 Extraction of Figures

We consider an existing algorithm, PDFFigure 2.0 [307], to extract various types of figures from the set of collected articles. Note that the PDFFigure 2.0 algorithm is designed to work on raster PDFs. This pruned the collection to 13.4K articles among the total of 20K articles, which are in the form of a raster PDF. We choose those articles to extract figures using PDFFigure 2.0. The algorithm analyzes the structure of each page by detecting captions, graphical elements, and chunks of body text and finally localizes the figures, and tables by reasoning about the empty regions within the text.

Category	Datasets			
	Deepchart [33]	Figureseer [34]	Revision [37]	DocFigure
Line graph	✓	✓	✓	✓ 9022
Natural image	-	-	-	✓ 3676
Tables	✓	-	✓	✓ 1899
3D object	-	-	-	✓ 1369
Bar plot	✓	✓	✓	✓ 1196
Scatter plot	✓	✓	✓	✓ 1138
Medical image	-	-	-	✓ 1128
Sketch	-	-	-	✓ 1105
Geographic map	-	-	-	✓ 1078
Flow chart	✓	✓	-	✓ 1074
Heat map	-	-	-	✓ 1073
Mask	-	-	-	✓ 1055
Block diagram	-	-	-	✓ 1024
Venn diagram	-	-	✓	✓ 889
Confusion matrix	-	-	-	✓ 811
Histogram	-	-	-	✓ 783
Box plot	-	-	-	✓ 605
Vector plot	-	-	-	✓ 576
Pie chart	-	-	✓	✓ 440
Surface plot	-	-	-	✓ 395
Algorithm	-	✓	-	✓ 392
Contour plot	-	-	✓	✓ 368
Tree diagram	-	-	-	✓ 360
Bubble chart	-	-	-	✓ 339
Polar plot	-	-	-	✓ 338
Area chart	-	-	✓	✓ 318
Pareto chart	-	-	✓	✓ 311
Radar chart	-	-	✓	✓ 309
Total samples	5K	30.6K	2K	33K

Table 5.1 Comparison of our DocFigure dataset with existing Deepchart [33], Figureseer [34] and Revision [37] datasets with respect to category labels and samples. The last column indicate the number of images in each class.

The figures present in the scientific documents can be a collection of various categories of sub-figures, we call them compound figures. PDFFigure 2.0 tool is unable to localize individual sub-figures in compound figure. To localize individual sub-figures in a compound figure, we use a similar concept described in [34] to iteratively decompose into sub-figures by identifying valid axis-aligned splits using the following criteria: (i) both resulting regions must have an aspect ratio between $1 : c_1$ and $c_1 : 1$ where $c_1 = 5$, (ii) ratio of the areas of the resulting regions must be between $1 : c_2$ and $c_2 : 1$ where $c_2 = 2.5$. The first criterion ensures that we avoid splitting it into extremely narrow sub-figures (this happens

due to accidentally split-off an axis or legend label). The second criterion enforces a weak symmetry constraint between the resulting halves (as sub-figures are all often approximately of the same sizes). Finally, we obtained $143K$ figures from the $13.4K$ articles. Due to the limitation of PDFFigure 2.0 and iterative decomposition of compound figures, 32% of total extracted figures are erroneous (i.e., the figures are over-segmented or they contain text regions).

5.4.3 Assignment of Category Labels to the Extracted Figures

Manual annotation i.e., assigning category labels to $96K$ figures, is a time-consuming and cost-ineffective job. Here, we propose an efficient way to assign category labels to many of sub-figures using the concept of incremental learning [308]. For this purpose, we develop a web-based annotation tool. Initially, we manually assign category labels to randomly selected 50 figures of each category, and we term this set of annotated figures as the `initial training set`. Therefore, a deep feature called FC-CNN (refer to Section 5.6.1) descriptor corresponding to manually annotated figures is obtained. We train a one-vs-rest linear support vector machine (SVM) using extracted descriptors corresponding to the `initial training set`. We also generate FC-CNN descriptors for the rest of the figures and calculate the similarity scores for belonging to each of these categories using the trained SVM. The annotation tool displays the top 100 figures of a particular class based on their similarity scores. The human annotators un-tick only those figures not belonging to the selected class and submit their recommendation. This way, we annotate more figures and create a `new training set`. We add additional examples that the annotator has selected and accept their annotations. We again train the SVM using both `initial and new training sets` and repeat this process until the recommended list has less than 50% images from the selected class.

Although the proposed annotation approach is efficient, however, it cannot annotate outlier figures in a particular category. The outlier figures are the figures which have less visual similarity (< 0.3) with the commonly occurring figures in a particular class. Those outlier figures make the dataset more complex. Only top similarity-scored (> 0.3) figures of a particular class are considered as training samples-while outlier figures of the same class are ignored during annotation (stage I). One possible solution to overcome this shortcoming is to include more diverse seed figures however; it is not always practically possible. Here, we propose a stage II annotation approach to include diverse (outlier) figures of a particular category in our dataset. In this approach, we select N figures from a set of figures and extract the FC-CNN descriptors corresponding to these figures [114]. We create a pair-wise similarity score of $N \times N$ matrix based on Euclidean distance between FC-CNN descriptors corresponding to N figures. Our tool shows 10 random figures from the N figures ($10 << N$), and the annotator can choose any random outlier figures. The tools display a set of figures which are similar to the chosen random figure. The annotator can refine the list by un-ticking and choosing the appropriate labels and submitting the annotations. This process is repeated until all figures are assigned their corresponding labels. Table 5.1 display the statistics of our dataset.

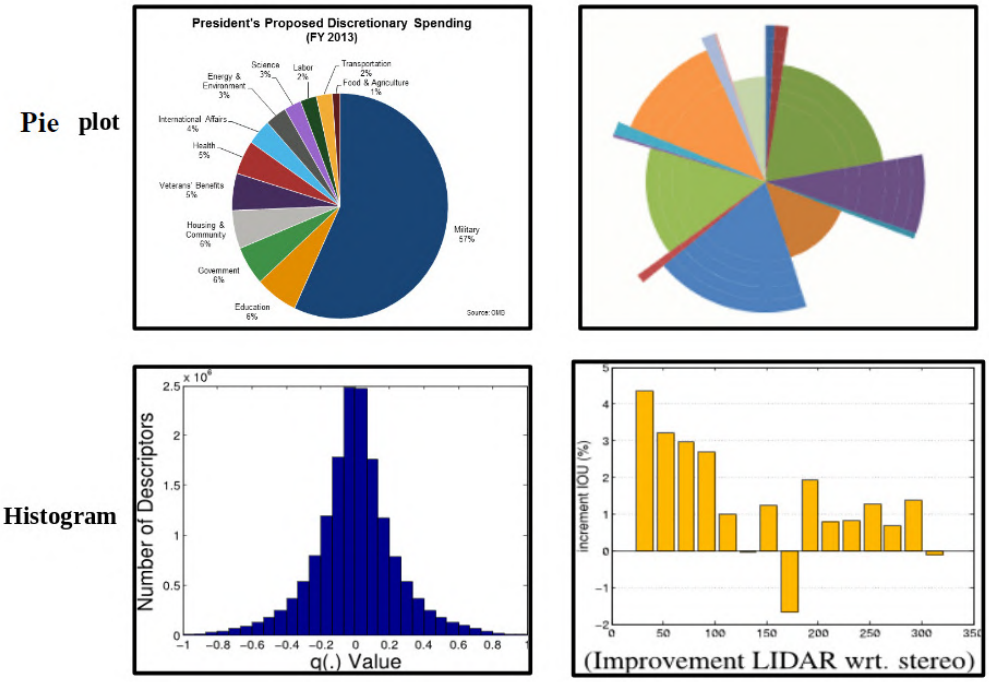


Figure 5.3 Intra-class dissimilarity in Pie chart and Histogram in `DocFigure` dataset.

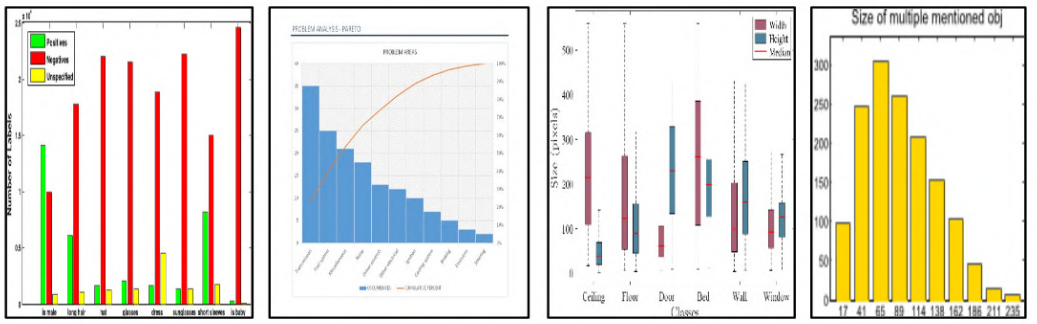
5.5 Complexity Analysis of `DocFigure` Dataset

Intra-class dissimilarity and inter-class similarity among various categories make the `DocFigure` dataset complex for classification tasks. Figure 5.3 shows the intra-class dissimilarity among the Pie chart and Histogram in the `DocFigure` dataset. This figure highlights that both these Pie charts are visually different from each other. Intra-class dissimilarity is also found in Histograms.

Figure 5.4 shows inter-class similarity among Bar plots, Pareto charts, Box plot and Histograms in the `DocFigure` dataset. From this figure, it is observed that the Bar plot, Pareto chart, Box plot, and Histogram are visually very much similar to each other. To better understand, we visualize the extracted feature vectors (FC-CNN and FV-CNN refer to section 5.6.1) using tSNE [301] method. From this figure, it is observed that all these three categories are overlapped to each other for both these feature: FC-CNN and FV-CNN spaces. From this figure, it is also observed that the visual similarity among Bar plot and histogram is more than similarity among Bar plot and Box plot and similarity among Box plot and Histogram. From the figure, it is also observed that FC-CNN feature is more effective for discriminating these three classes than the FV-CNN feature.

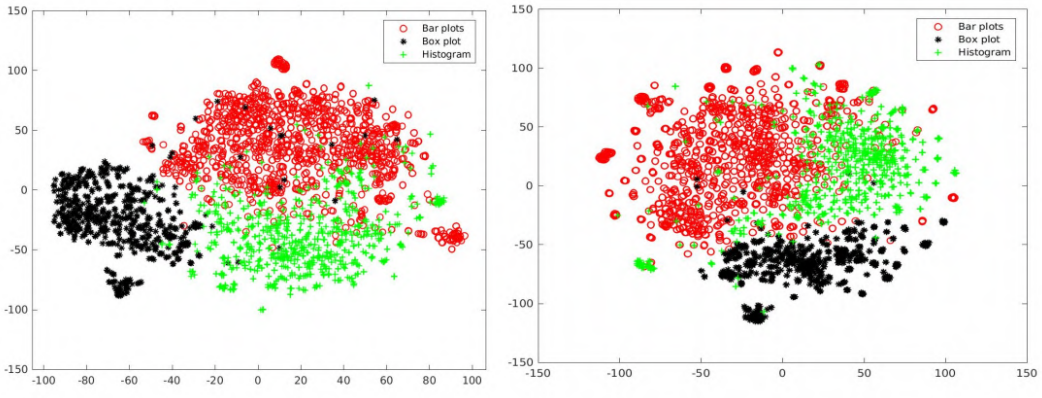
5.6 `DocFigure`: Proposed Baseline Approaches

We propose three baselines to validate our generated dataset `DocFigure` on the document figure classification task. Features play an important role in classification. It is already well established that



Bar plot Pareto chart Box plot Histogram

Figure 5.4 Inter-class similarity among Bar plot, Pareto chart, Box plot and Histogram categories of `DocFigure` dataset.



(a) (b)

Figure 5.5 (a) and (b) are T-SNE visualization of FC-CNN and FV-CNN features of Bar plot, Box plot and Histogram images in `DocFigure` dataset, respectively.

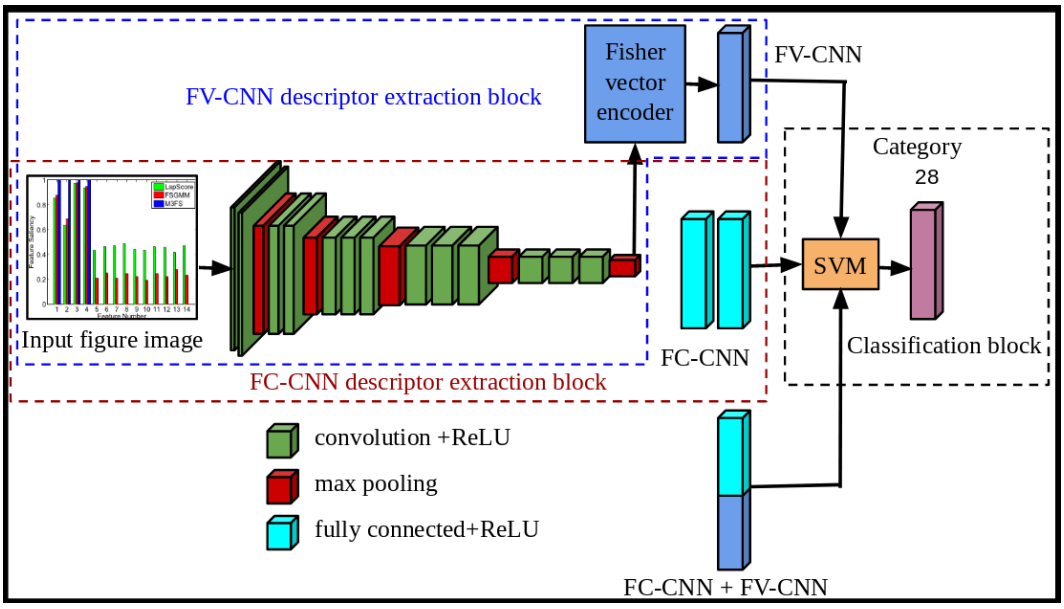


Figure 5.6 Basic framework for the proposed three baseline approaches. The red dotted rectangle corresponds to FC-CCN features extraction block, The blue dotted rectangle indicates FV-CNN features extraction module and Black dotted rectangle corresponds to the classification module (best viewed in color).

the FC-CNN descriptor, which is obtained by extracting features as output of the penultimate Fully-Connected (FC) layer of a CNN, have great success in image classification tasks [103, 104, 106]. Cimpoi et al. [114] proposed FV-CNN descriptors which are obtained by Fisher Vector pooling of a CNN filter bank for semantic segmentation task. We consider both these descriptors as features for the figure classification task.

Figure 5.6 displays the basic outline of our three baseline approaches. Each of these approaches consists of two basic modules: feature extraction and then classification. We consider deep features (FC-CNN) and deep texture features (FV-CNN) extracted from the feature extraction module and a combination of both these features to represent each figure image. One-vs-rest SVM is chosen as a classifier to assign category labels to the figure image in the classification module.

5.6.1 Feature Extraction Module

It takes a figure image as an input and extracts deep features: FC-CNN as object descriptor and deep texture features: FV-CNN as texture descriptor as output. We consider pre-trained VGG-V [193] to extract these features. Here, we discuss both these descriptors.

5.6.1.1 Object Descriptor:FC-CNN

The FC-CNN descriptor is obtained by extracting features as the output of the penultimate Fully Connected (FC) layer of Convolutional Neural Network (CNN), which takes figure image as input. The

red dotted rectangle in Figure 5.6 indicates the module for extraction of FC-CNN. This extracted feature can be considered as an object descriptor because the fully connected layers allow FC-CNN to capture the overall shape of the object contained in the region.

5.6.1.2 Texture Descriptor: FV-CNN

The FV-CNN descriptor introduced by Cimpoi et al. [114] based on texture descriptor [183] using Fisher Vector (FV) encoding technique. We perform FV encoding on the output of the last convolutional layer of the convolutional neural network. Since the fully connected layer of the network is not involved in the FV-CNN feature generation, images with various sizes can be used to generate the FV-CNN features. Different from FC-CNN, FV pools local features densely within the regions by removing global spatial information. Therefore, this feature describes textures rather than objects. FV is computed on the output of last convolution layer of CNN. The blue dotted region in Figure 5.6 indicates the FV-CNN feature extraction module.

5.6.1.3 Combination of FC-CNN and FV-CNN

We also concatenate both the features FC-CNN and FV-CNN to represent figure image.

5.6.2 Classification Module

We use a one-vs-rest linear support vector machine (SVM) [111] to assign category labels to the figure images. The black dotted rectangle in Figure 5.6 specifies the classification module. SVM is trained with each of three different representations (FC-CNN, FV-CNN and FC-CNN+FV-CNN) corresponding to the training set containing figure images. Finally, trained SVM is used to assign category labels to figure images of the test set.

5.7 Experiments

5.7.1 Implementation details

The FC-CNN and FV-CNN features corresponding to document figure images are extracted using a pre-trained VGG-V model [193]. This network architecture produces the FC-CNN feature with 4096 dimension and FV-CNN with 512-dimensions. In the FC encoding, 16 Gaussian components are used on the 512-dimensional convolution features, resulting in 16K-dimensional FV-CNN feature. In order to accommodate the different image scales, we calculate the FV-CNN feature after re-scaling the image by factors $2^s,\ s = -3, -2.5, ..., 1.5$ (for efficient calculation, we select the scale for which the number of image pixels in the range 30 to 1024^2).

5.7.1.1 Learning details.

The descriptors FC-CNN, FV-CNN and FC-CNN+FV-CNN corresponding to figure images are classified using a one-vs-rest SVM classifier. We normalize each descriptor using L_2 before classification and set miss classification weight $C = 1$. After normalization, C has minimal effect on SVM performance.

Furthermore, to improve SVM performance, we re-calibrate the SVM score after training by scaling the weight vector and bias such that the median scores of the negative and positive training samples for each class is mapped to -1 and 1, respectively.

5.7.2 Quantitative Results Analysis

Results obtained using our proposed three baseline approaches are summarised in Table 5.2. We obtained the best classification accuracy while both FC-CNN and FV-CNN descriptors were concatenated to represent the figure images. Bold values indicate the best obtained results in Table 5.2. We observed that FV-CNN is more effective than FC-CNN for the document figure classification task (except 3D object, Algorithm, Bar plot, Box plot, Flow chart, Heat map, Histogram, Medical image, Pie chart and Polar plot) as it represents texture rather than object shape. The use of the FV-CNN descriptor improved average classification accuracy by 1.84% over the FC-CNN descriptor. While combination of both FC-CNN and FV-CNN improves the classification accuracy over individual features, FC-CNN and FV-CNN for all categories except Bar plots, Bubble charts, Contour plots, Geographic maps, Radar charts, Surface plots, Tables and Venn diagrams. We also noticed that the use of concatenation of FC-CNN and FV-CNN improved the average classification accuracy by 3.94% and 2.10% over individual use of FV-CNN and FC-CNN, respectively. With this experiment, we concluded that the deep texture descriptor (FV-CNN) is better than the shape descriptor (FC-CNN) for a few classes.

Labels	FC-CNN	FV-CNN	FV-CNN +FC-CNN
3D objects	98.24%	94.73%	**98.53%**
Algorithm	**93.81%**	91.75%	**93.81%**
Bar plots	**93.97%**	91.97%	93.64%
Box plot	91.39%	88.07%	**92.05%**
Flow chart	92.53%	91.04%	**97.01%**
Heat map	99.25%	95.89%	**99.62%**
Histogram	**94.89%**	88.26%	**94.89%**
Medical images	97.87%	92.55%	**98.93%**
Pie chart	91.66%	89.81%	**94.44%**
Polar plot	**85.71%**	78.57%	**85.71%**
Area chart	84.61%	91.02%	**92.30%**
Block diagram	97.26%	97.65%	**98.43%**
Bubble Chart	80.95%	**91.66%**	90.47%
Confusion matrix	85.22%	89.65%	**93.10%**
Contour plot	59.34%	**74.72%**	72.52%
Geographic map	88.59%	**95.81%**	95.43%
Graph plots	98.49%	98.84%	**99.33%**
Mask	**99.23%**	**99.23%**	**99.23%**
Natural images	98.04%	98.25%	**99.23%**
Pareto charts	87.17%	96.15%	**97.43%**
Radar chart	78.94%	**86.84%**	85.52%
Scatter plot	90.14%	91.19%	**93.66%**
Sketches	95.65%	96.37%	**98.18%**
Surface plot	76.76%	**89.89%**	88.88%
Tables	97.25%	**98.73%**	97.67%
Tree Diagram	67.04%	68.18%	**70.45%**
Vector plot	79.86%	81.94%	**86.80%**
Venn Diagram	87.03%	**93.51%**	93.05%
Average	88.96%	90.80%	**92.90%**

Table 5.2 The class-wise accuracy of 28 classes in our proposed dataset `DocFigure` using shape feature (FC-CNN), texture feature (FV-CNN) and combination of both (FC-CNN+FV-CNN). The labels written in italics are more discriminative in shape feature than texture feature.

5.8 Lecture Slide Deck (LecSD) Dataset

We present a very large-scale, one-of-a-kind lecture slide dataset, namely the LecSD. This dataset's image and associated annotations can be downloaded from our project website upon acceptance of this paper. The slide images of the LecSD are harvested from the web[1] for a popular computer science and engineering course, namely *Data Structures*. We used the following popular sub-topics to search slide decks: *a) Arrays and Structures, b) Stacks and Queues, c) Lists, d) Trees, e) Graphs, f) Sorting, g) Hashing, h) Heap Structures, i) Search Structures, j) Algorithms, k) Stacks and Queues, l) Queues,* and *m) Binary trees*. By cleaning the collected slide image by removing duplicate slides and slides with no meaning, we obtained 1700 slide decks with around $54K$ slide images. We split the data into train, validation, and test sets of $30K$, $10K$, and $10K$ slide images, respectively. In addition, we collected and manually annotated 4000 slide images from the topic of *Computer Networks* and *Optimization* to demonstrate the generalizability of the proposed model.

5.8.1 Annotations

We obtain annotations for our dataset to enable retrieval of slide images and their components such as figures, tables, and equations using multimodal queries, including natural language text and drawing. We restrict our manual annotation to only the evaluation (test) and validation set since manual annotations of queries for building a large system are time-consuming, cumbersome, and not scalable. We automatically annotate the training data using state-of-the-art slide segmentation, figure classification, and OCR modules.

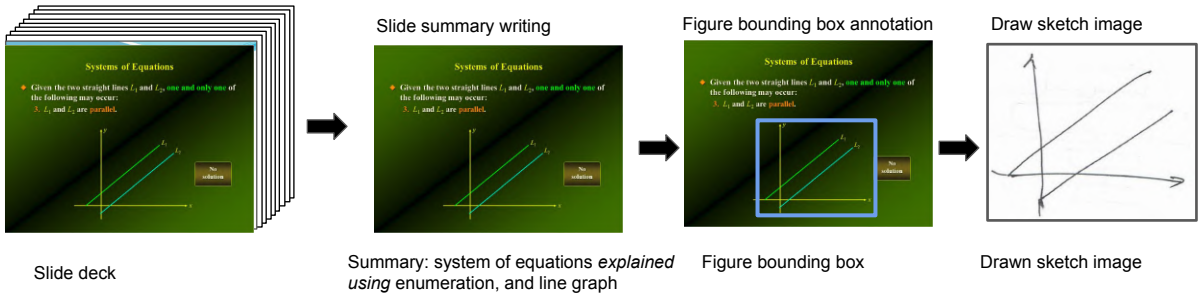


Figure 5.7 We present the LecSD towards developing a benchmark for retrieving educational content, specifically lecture slides from computer science topics. Here, we show our proposed annotation pipeline. First, annotators were asked to write a summary of the slides from the collection of slide image decks. Then, annotate the bounding boxes for the figures, and finally, draw a sketch image corresponding to the annotated figures. (Best viewed in color).

[1] https://slideplayer.com/

5.8.1.1 Manual annotation

We generate organic queries for slide image retrieval with the help of five annotators. To obtain manual annotations for the test and validation set of our dataset, we provide slide images to the annotators and ask them to write a brief sentence about a given slide image. This brief sentence serves as a summary query for our dataset. Further, if figures were present in the slide, annotators were asked to draw the corresponding sketch of the figure on a paper. This paper is scanned, and the cropped sketch region is used as a sketch query for the slide. Figure 5.7 shows the annotation pipeline for a single slide image. We provide a modified version of the VGG image annotation tool [309] to annotators for annotating the slide image summary and figure regions. In order to overcome the annotation bias [310], we used chatGPT [311] to generate paraphrased sentences of written summary. Sample drawn sketches and written summaries are shown in Figure 5.8.

5.8.1.2 Automatic annotation

The primary goal of automatic summary annotation is to train the language model to retrieve slide images and give an organic query. We conducted a study on organic queries to retrieve slide images and concluded it as follows: i) we noticed that the keyword specific to a slide most frequently occurs in the title of the slide image. ii) the keywords can also occur in enumerations or paragraphs for the slides with the slide image having no titles or the title of common words such as overview, conclusion, methods, and problem. iii) the summary can also contain the list of logical regions such as enumeration, paragraphs, tables, equations, and various figure classes such as line graphs, bar charts, photographs, etc. iv) the logical region name need not be consistent in the summaries. As an example, the enumeration can be mentioned as bullet points. Hence, we designed the automatic slide summary as a predefined sentence structure, as $\underline{T}$ explained using $\underline{C}$. Where $\underline{\mathrm{T}}$ is the OCR text obtained from slide title, enumeration, or paragraph regions. $\underline{\mathrm{C}}$ is the list of logical regions such as enumeration, paragraphs, tables, equations, and figures. We randomly replace the logical region names with their synonyms. The slide layout segmentation model [56] is used to identify the regions. To identify the type of figures present in slide images, we use a trained model with `DocFigure` [46], having 28 various types of figure classes. The sketches of figures in the slide image are automatically created using Photo-sketching [312]. The Photo-Sketching model is designed to generate contour drawings and boundary-like drawings that capture the outline of the visual scene. Hence, the model is well-suited for creating sketches of document figures. Figure 5.8 shows the sample sketches created by the Photo-sketching model. First, we identify the figure regions using the layout segmentation model and create sketches using the Photo-sketching model.

We manually extract text, draw the layout, and identify figure types from 100 slide images to evaluate the quality of automatic extraction of text, layout, and figure class. To extract the text in slide images, we use Google Lens OCR with a word error rate of 4.63%. The layout segmentation of slide images is performed using CSSNet [56] trained on SPaSe [8] and WiSe [7] dataset and obtaining the MIoU of 56.4%. Finally, the figure classes are identified by training the Multi-feature head model [313] using

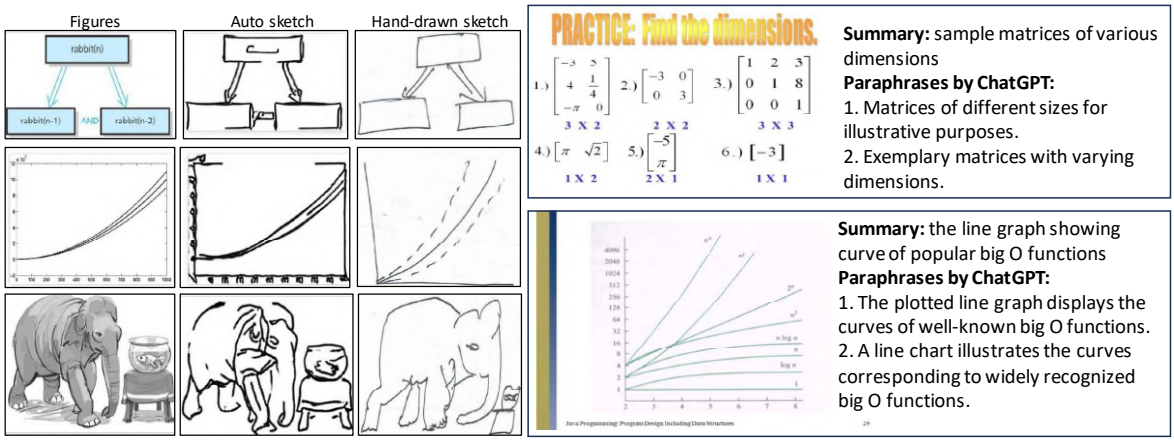


Figure 5.8 Sample figures in LecSD dataset. The first column shows cropped figure regions from the slide image. The second column of sketches is generated using Photo-Sketching [312]. The third column shows the manually drawn sketches. The last column shows the slide images and its manual summary and its two paraphrased sentence generated using chatGPT [311].

DocFig [46] dataset and obtained an accuracy of 97.85%. After annotation, the train, validation, and test have 5607, 1557, and 1487 slide images with figures, respectively.

5.8.2 Dataset analysis

To analyze the distribution of topics in the dataset, we extracted trigram keywords from the text data of each slide image using KeyBERT [314] and obtained $252K$ keywords from the dataset. Figure 5.9 shows the sunburst of common keywords, and the data structure topics such as 'tree', 'list', 'heap', 'queue', and 'stack' are fairly distributed in the dataset. The inner-circle keywords occur at least $10K$ times and the outer-circle keywords occur more than 50 times in the entire dataset.

To analyze the collected slide images, we use two approaches: a) an Optical Character Recognizer (OCR) with text properties and b) layout-based image segmentation. In the first approach, we utilize off-the-shelf Google lens OCR engine[2] to extract the text information from the slide images. Further, we categorize the text in the slide based on font size. Figure 5.12(a) shows the distribution of words with various font sizes in the dataset. The font size is the height of a word, and the number of pixels represents the height. As shown in the figure, we divide the histogram into four regions as follows– S: small, M: medium, H: high, and L: large font size with a height of $(0,\ 15]$, $(15,\ 30]$, $(30,\ 45]$, and > 45 pixels respectively and create the word cloud images using the words in each region. Word clouds provide a simple and effective means to communicate the most frequent words of the documents visually. We further analyzed our dataset – LecSD, using image properties such as the number of color channels in the image. In the analysis, first, we sorted out the slide images based on their RGB channels and obtained only 6261 grayscale images out of $54K$ slide images. Generally, grayscale slide images have a

[2]https://cloud.google.com/vision/docs/ocr

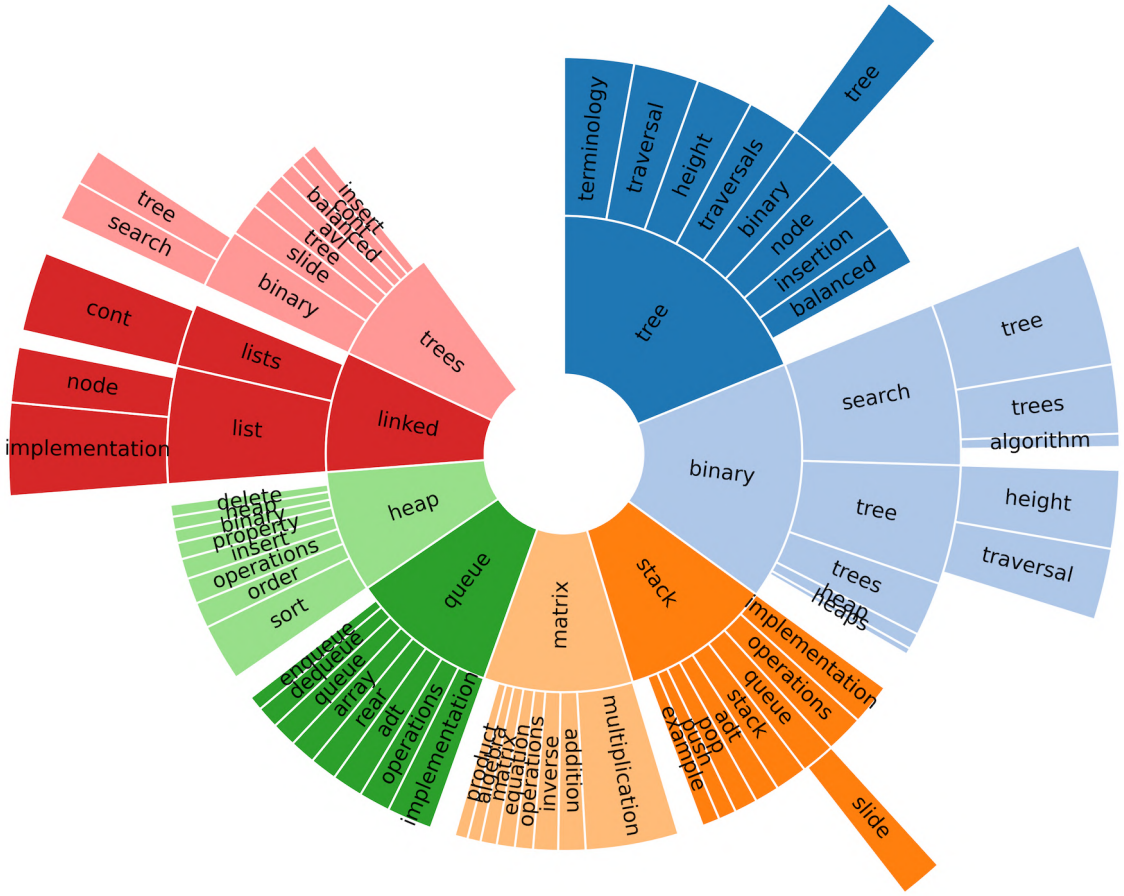


Figure 5.9 The topics and sub-topics in LecSD dataset visualize using a sunburst of trigram keywords extracted from the text data of each slide images using KeyBERT [314].

less complex layout design than color slide images. Next, we studied the layout components and types of figures used in the slide images. Figure 5.10 shows the frequency of occurring text layout regions, and Figure 5.11 shows the types of figures in the slide dataset. Figure 5.10 shows that the 'Enumeration' region has a higher frequency, and the 'Footnote' has the lowest frequency. Figure 5.11 shows the most frequently occurring document figure is the 'flowchart'.

Figure 5.12(b) shows the word cloud of the four regions. The word cloud of regions M, H, and L have common words, and the word cloud image of the S region has different words related to the copyright and publication details. Figure 5.12(c) shows the typical example of four regions of text appearing in a slide image. The L, H, and M regions are 'title', 'section', and 'paragraph/list' items. Hence, it shares common words on the topic of *Data structures*. However, the small font size text belongs to represent 'affiliation', 'date', 'slide number', 'press titles', 'footnote', and 'website'. Hence, the words in the S region's word cloud differ from those in other regions.

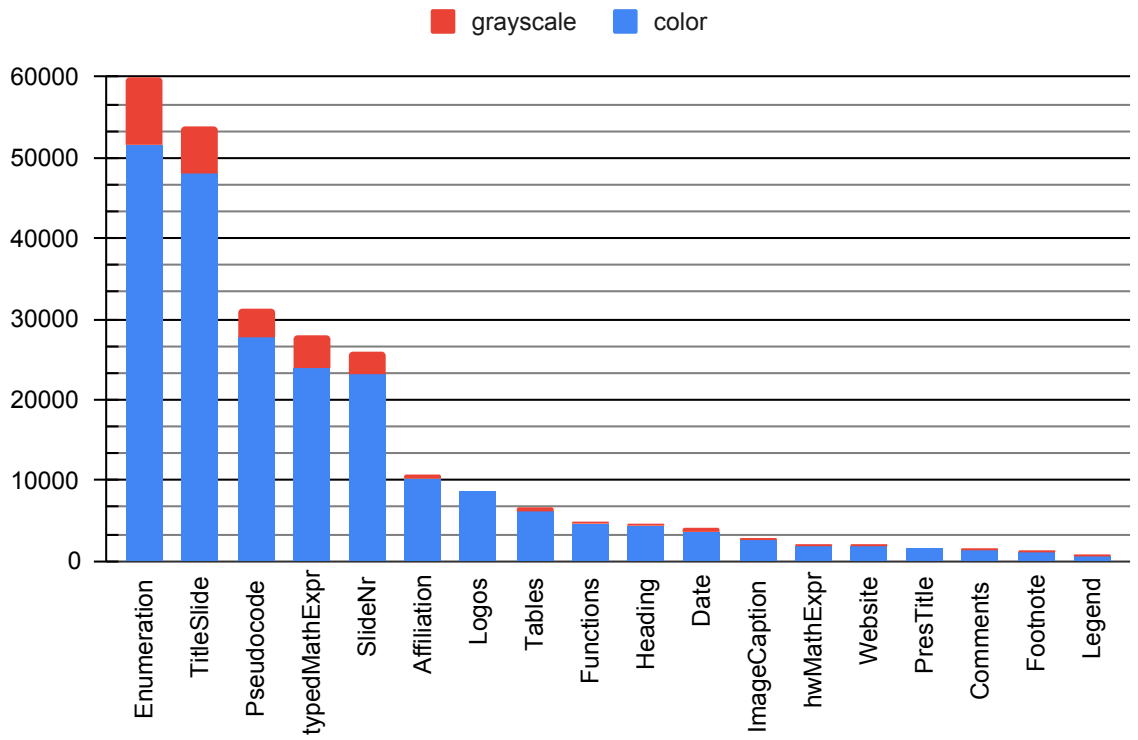


Figure 5.10 Distribution of text logical labels in LecSD dataset.

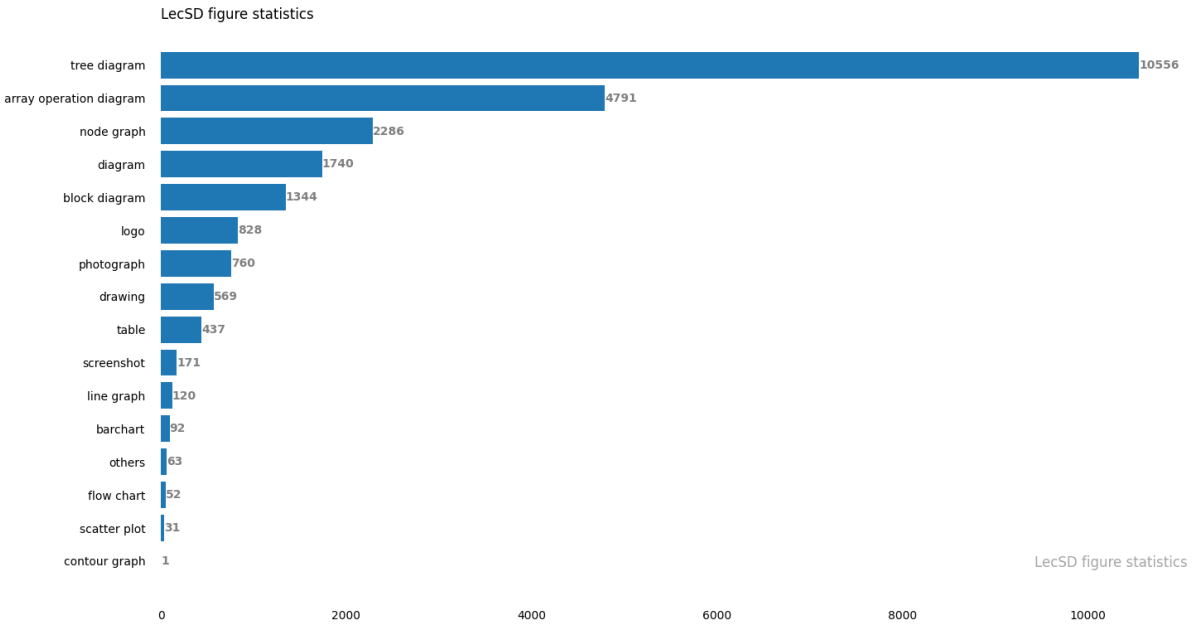


Figure 5.11 Distribution of figure logical labels in LecSD dataset.

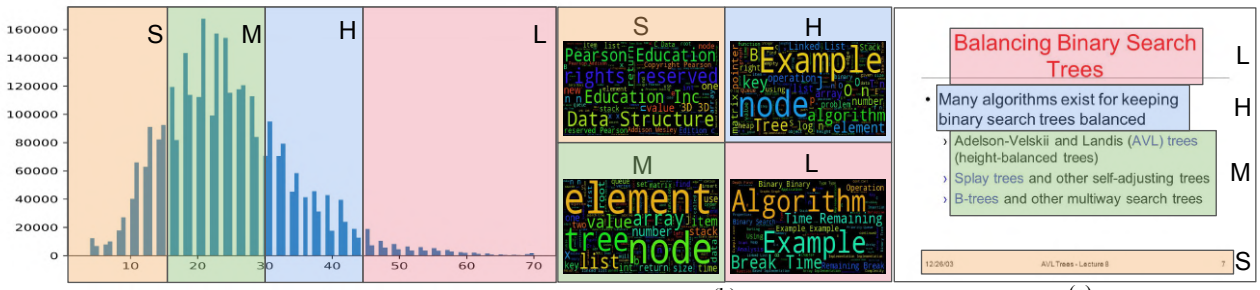


Figure 5.12 Visualize various font size text in the slide images (a) histogram of words with various font sizes and divide the histogram into four regions: S: small, M: medium, H: high, and L: large font size. (b) The word cloud images of words from S, M, H, and L regions. (c) A typical example of text with various font sizes in a slide image (best view in color)

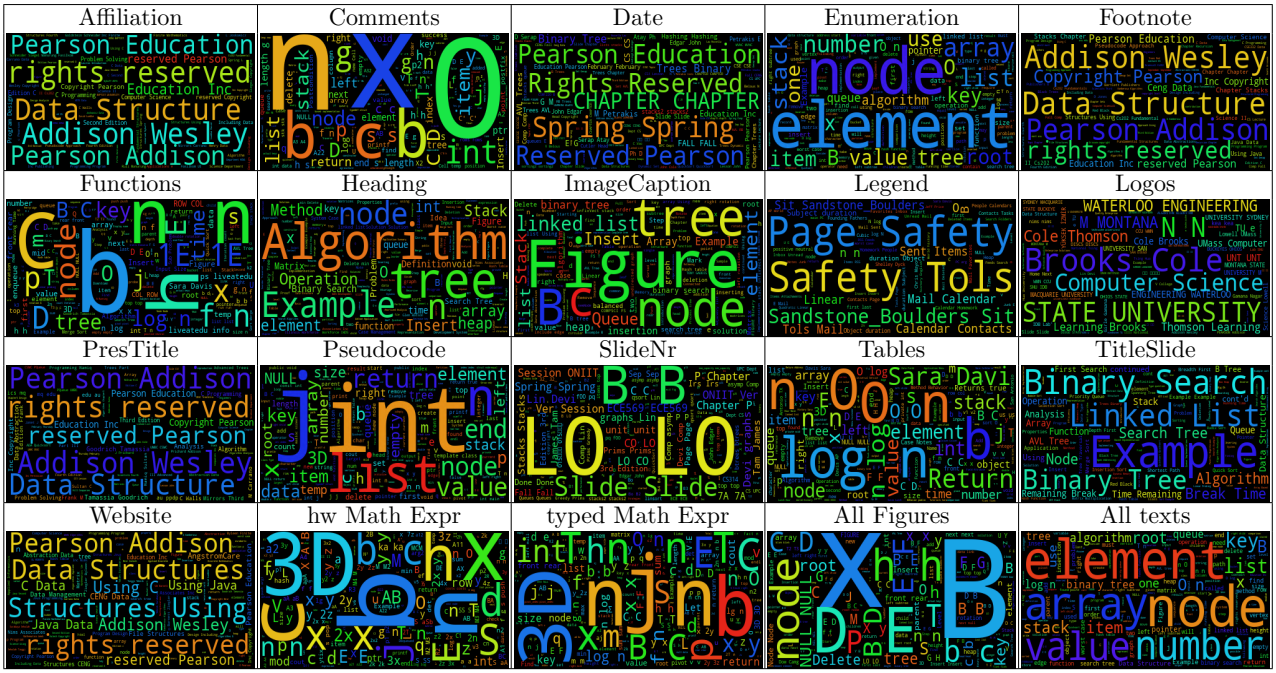


Figure 5.13 The word cloud image of the text extracts from the various logical regions in the slide dataset. The odd rows indicate the logical region, and the figure below is its word cloud image.

We extract the text information from the slide, combine the text from similar logical regions and plot its word cloud [315], as shown in Figure 5.13. By analyzing Figure 5.13, the layout segmentation has higher accuracy in segmenting most logical regions except 'Date', and 'Slide number'. The word cloud of 'Affiliation', 'Pres title', 'Website', and 'Footnote' have high similarities, indicating a high confusion in segmenting these regions. We also plot the word cloud of all texts in the slide, and it is highly similar to the word cloud of the 'Enumeration' text.

5.8.3 Slide Image summary

We used a modified version of VIA annotation software [309] and a screenshot of the web-based annotation tool is shown in Figure 5.18. Table 5.4 and 5.5 shows sample slide images and their manual summary, ChatGPT [311] generated paraphrase, Automatic summary, Synonym-based paraphrased sentences.

In the synonym-based paraphrase generation, first, we Identify the slide topic T and semantic regions C. We replace each semantic region with its corresponding synonyms. Some of the sample synonyms are shown in Table 5.3. We also randomly change the structure of the sentence. Half of the paraphrased sentence with structure T explained using C, and half with C is used to explain T.

block diagram	**Diagram**	**enumeration**	**equation**	**Line graph**	**paragraph**
Chart of relationships	Sketch	List items	Formula	Line chart	Block of text
Conceptual diagram	Artwork	Itemized points	Algebraic expression	Trendline	Section
Schematic diagram	Illustration	Key details	Numeric expression	Graph	text Segment
Structural diagram	Rendering	bullet points	Mathematical formula	Chart	Passage

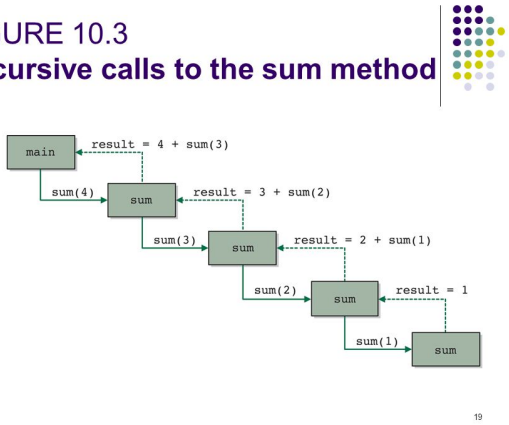
Table 5.3 A sample synonyms of commonly occurring words such as 'block diagram', 'sketch', 'enumeration', 'line graph', and 'paragraph' in the summary of slide image

5.8.4 Figure bounding box annotation

We manually annotate figure bounding boxes in the slide image. A screenshot of the web-based annotation tool is shown in Figure 5.14.

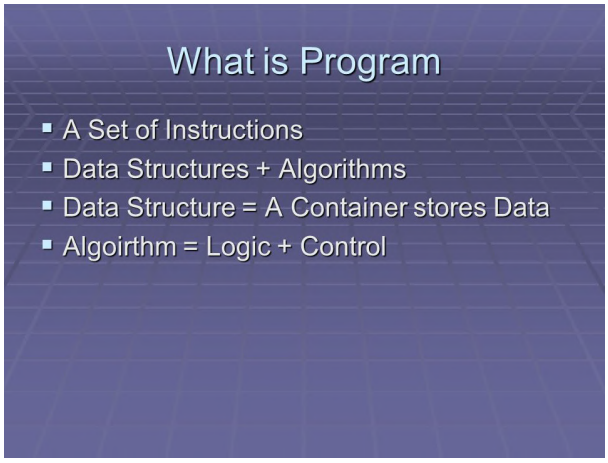
5.8.5 Figure sketches drawing

Our annotation team drew the annotated figures on a specially designed A4 paper. A screenshot of the web-based annotation tool is shown in Figure 5.15.



Manual Summary: A block diagram depict the recursive calls to the sum methods.
ChatGpt: A block diagram illustrates the repetitive invocations of the sum function.

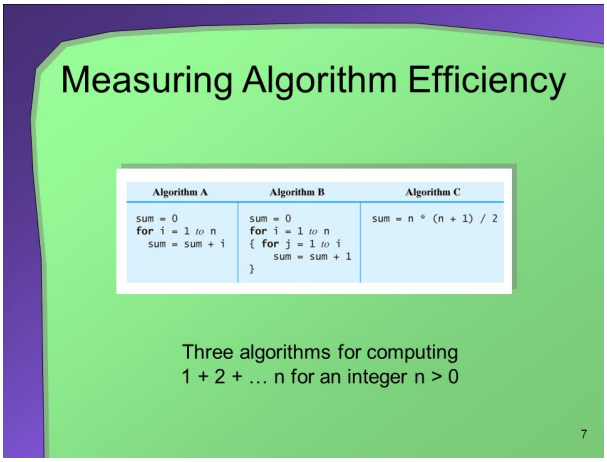
Manual Summary: Recursive calls to the sum method explains with a block diagram
Synonym based paraphrase: Abstract code Used for describing Recursive calls to the sum method.



Manual Summary: The definition of program explains using enumeration.
ChatGpt: The program's definition elucidates the utilization of enumeration.

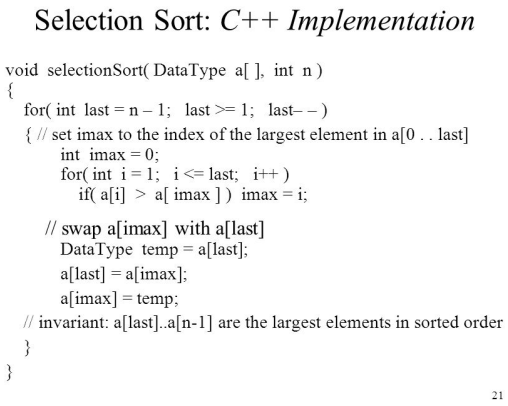
Automatic Summary: What is Program explains using Enumeration
Synonym based paraphrase: Itemized points Employed to elucidate the What is Program

Table 5.4 Showing sample slide images and its manual summary, ChatGPT generated paraphrase, Automatic summary, Synonym based paraphrased sentences



Manual Summary: three algorithms for calculating sum of n integers given in a table.
ChatGpt: Three methods for computing the sum of a set of n integers provided in a table.

Manual Summary: Measuring Algorithm Efficiency explains using a table and Paragraph
Synonym based paraphrase: Measuring Algorithm Efficiency Describes with tables, and text Segment.



Manual Summary: The code of Selection sort implementing using C++.
ChatGpt: The C++ implementation of Selection Sort code.

Manual Summary: Selection Sort: C++ Implementation explains using a Pseudocode
Synonym based paraphrase: Selection Sort: C++ Implementation Depicts using algorithm.

Table 5.5 Showing sample slide images and its manual summary, ChatGPT generated paraphrase, Automatic summary, Synonym based paraphrased sentences

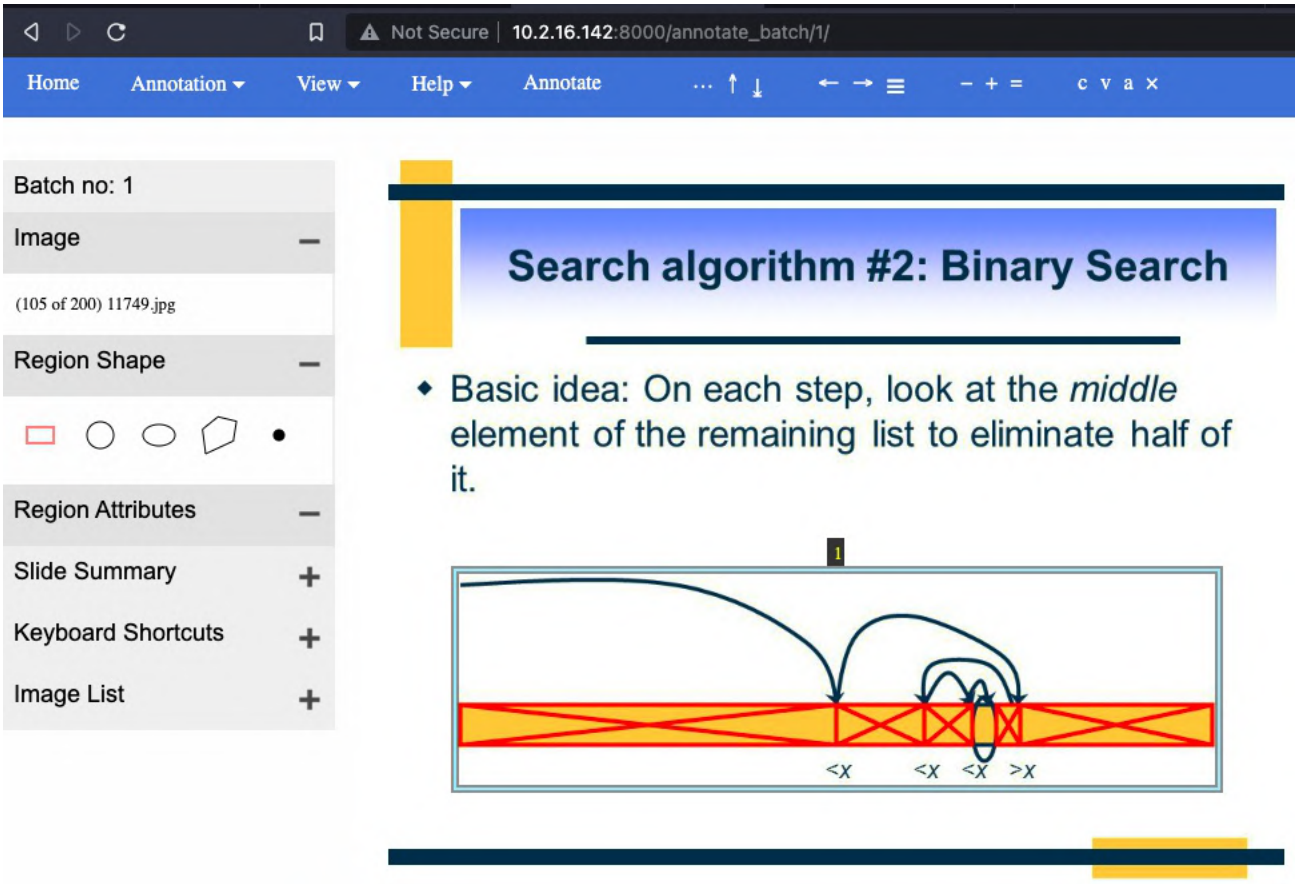


Figure 5.14 Figure bounding box annotation using a web-based annotation tool.

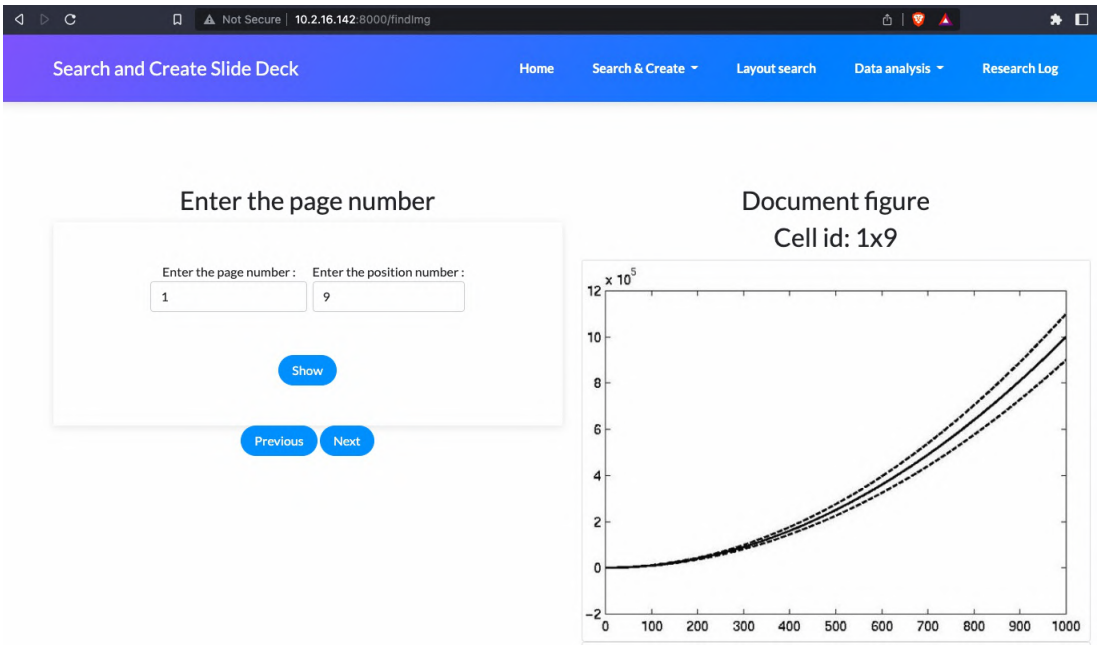


Figure 5.15 The web-based annotation tool shows the figure and its cell id to sketch the image.

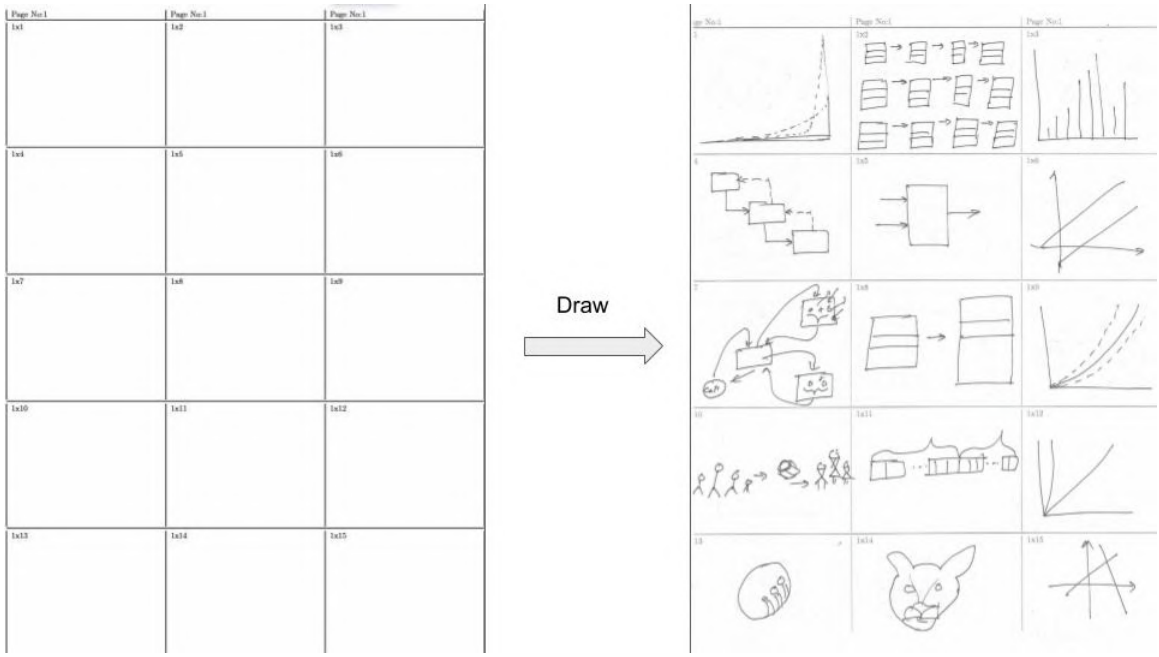


Figure 5.16 The left side shows the A4 paper canvas with 15 table cells with unique cell id. The right side shows the A4 paper canvas after drawing the figure image.

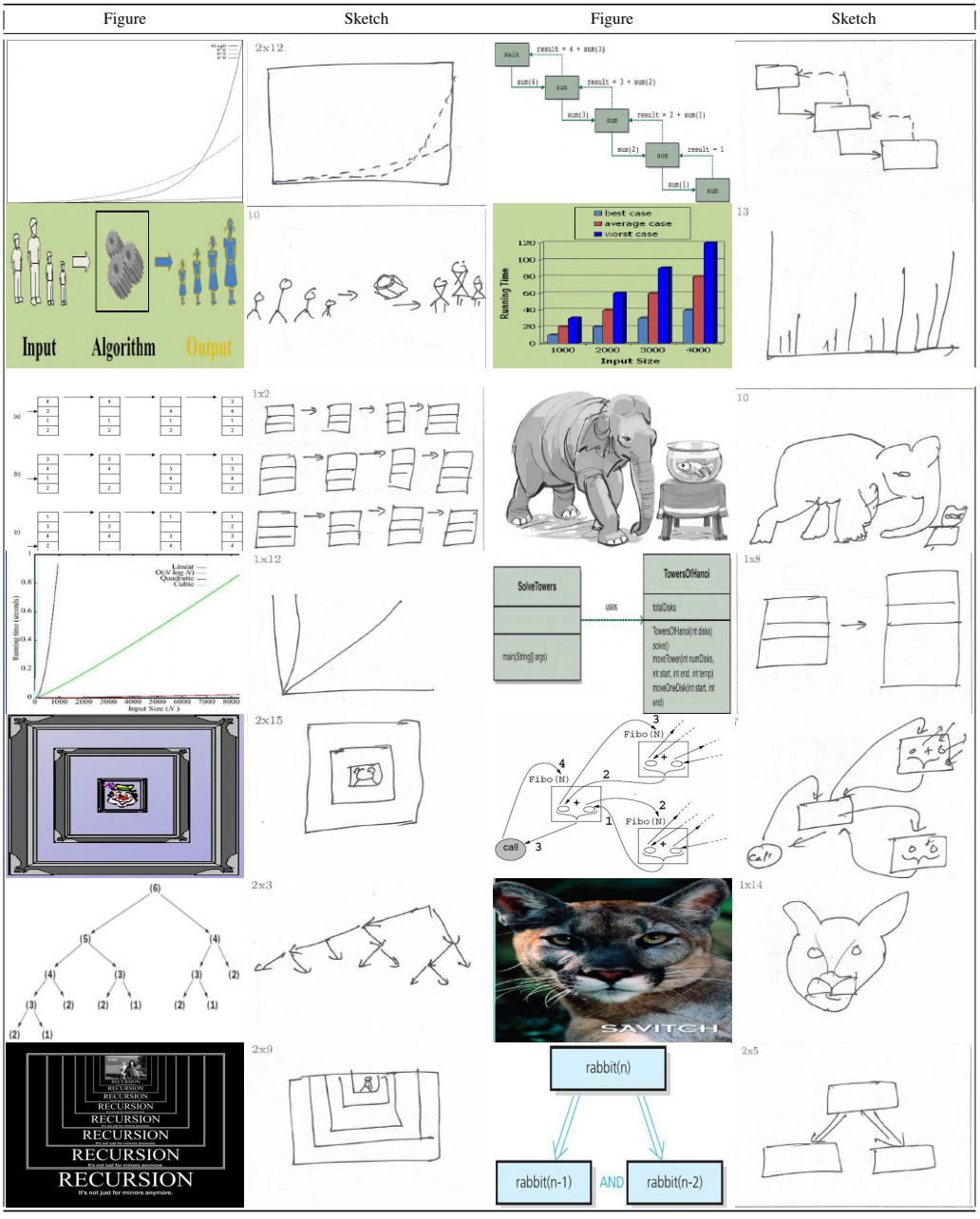


Figure 5.17 Sample slide image figures and their corresponding drawn sketch of our dataset. The figure columns show the figure annotated from the slide images and the right-side Sketch columns shows its hand-drawn sketches.

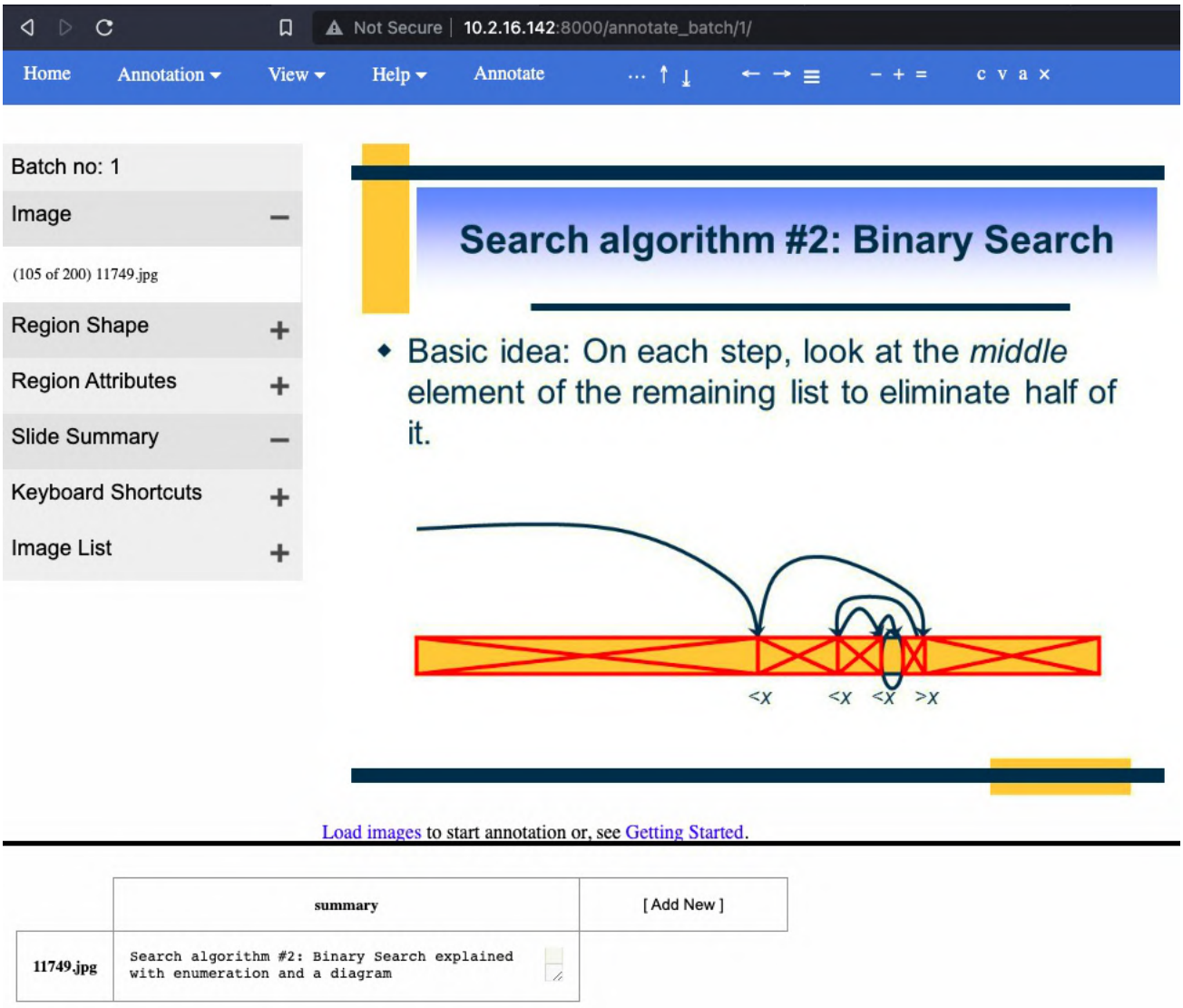


Figure 5.18 The slide image summary writing using a web-based annotation tool.

Chapter 6

Applications

Building on the research presented in the previous chapters, this chapter discusses three new document analysis applications. The applications centered around document layout segmentation. The first application addresses the common challenge of adapting document templates for reading on various electronic devices, such as mobile phones, tablets, and desktop computers. These templates must often be adjusted adequately, necessitating cumbersome zooming and panning. To overcome this issue, we propose a novel approach for seamlessly changing document templates, optimizing the reading experience across different devices.

The second application is a unique slide narration system. While slide presentations have long been an efficient teaching tool, visually impaired students often need help accessing and engaging with this teaching format. Recognizing this limitation, we have devised a solution to generate audio descriptions that align with the content of each slide, thereby enhancing the learning experience for blind and visually impaired individuals.

The third and final application focuses on accessing required slide images from the Massive Open Online Courses (MOOCs). These platforms provide easy access to many educational materials, including lecture slides. However, the sheer volume of information available demands effective search mechanisms. We introduce the Lecture Slide Deck Search Engine (LecDeckSearch Engine) to tackle this challenge. This innovative search engine supports natural language queries and even hand-drawn sketches, enabling users to perform searches across an extensive collection of slide images on computer science topics.

6.1 Template Transfer

Smartphones are now successfully used for reading documents due to advancements in digital media processing and visualization capabilities. Algorithms for reflowing markup documents (e.g., HTML) across displays of varying resolutions have reached the level of standards. However, when images need to be viewed on various displays, one must manually adjust their location and change the zoom levels. For a class of natural images, content-aware seam carving methods [316, 317] are attractive. They find perceptually less critical regions to be removed/cropped, leading to a lossy but visually appealing

image. However, even a small information loss could be severe for the display/reading of document images. A document image has textual content recognizable by OCRs and graphics components that complement the text's information. For the rest of the paper, we used to document and document images interchangeably.

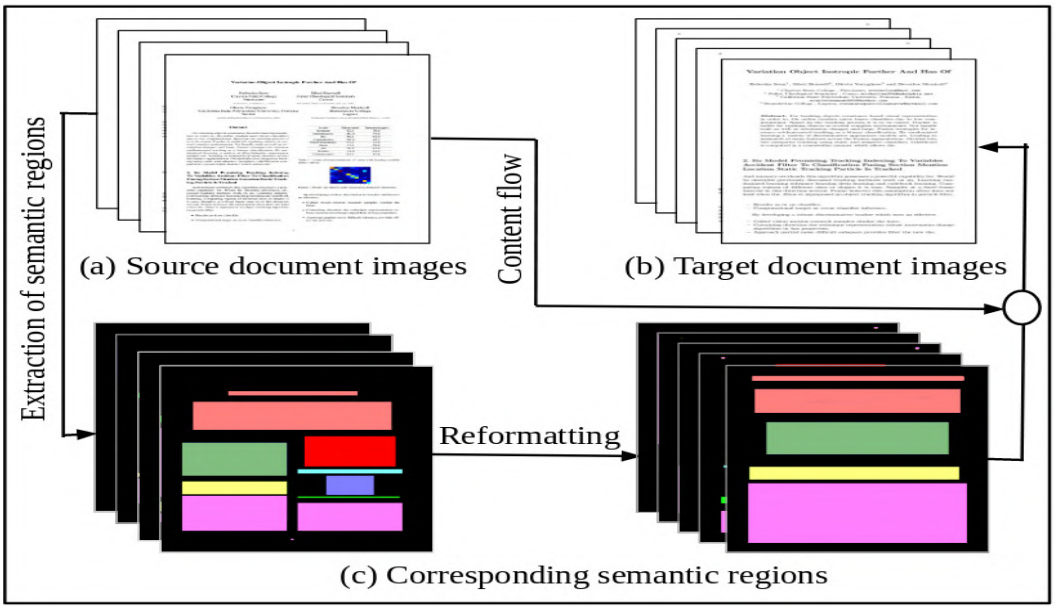


Figure 6.1 Is it possible to reformat a sequence of images formatted in CVPR style to that of ECCV style at the image level? We believe this exploration opens up an important class of image reflowing/resizing algorithms that can support various displays.

We define a document reformatting problem as reflowing of one or more input document images in one layout/style to another layout/style. In a way, this is an image-to-image translation problem with goals beyond the visual appeal. According to Isola *et al.* [262], image-to-image translation is the task of translating one representation of an image to another by learning pixel-to-pixel mapping. In most problems formulated in this setting, image-to-image or pixel-to-pixel correspondence exists. Our interest is in realigning the areas/regions according to a set of constraints available. As an example, we also demonstrate how a collection of document images (multiple pages) formatted in CVPR (or IEEE two columns) style can be transferred to an ECCV single column (Springer) style image by working at the level of images. Figure 6.1 depicts this problem. We formulate the document reformatting problem as a sequence-to-sequence translation problem inspired by machine translation literature [318]. Regions from the pages of the source document are considered as the input sequence. In contrast, the predicted sequence corresponds to the regions' location and sizes from the target document. Our method first extracts interest regions with a semantic segmentation algorithm and then learns the style transfer with the sequence-to-sequence model. Semantic segmentation in document images is still in the nascent stage. There are not enough labeled data sets available for the training of the popular deep models. We intro-

duce two data augmentation schemes to address this. The performance of our semantic segmentation algorithm is shown to be superior to the state-of-the-art.

Contributions of this work are :

- We propose a framework to reformat a source document in one layout style to a target document with a different layout style.
- Our method is validated on a multi-page document image setting.
- Our document image segmentation model reports superior results compared to the state-of-the-art [319].
- To address the problem of data scarcity in training deep models for semantic segmentation of document images, we introduce data augmentation approaches that are shown to be very useful.

6.1.1 Template transfer formulation

We formulate a document image reformatting task as a sequence-to-sequence translation problem inspired by a machine translation network [318]. In our case, segmented document images corresponding to a source document are considered as input sequences, whereas predicted sequence corresponds to segmented document images of the target document. Finally, the content of the segmented regions of the source document is transferred to the corresponding predicted regions to obtain the target document. Each of these modules is discussed in the following subsections.

6.1.1.1 Document Image Reformatting Module

Our document image reformatting module, as shown in Figure 6.2, reformat a source document in a given style into a target document with another style. This module consists of three sub-modules: (i) image encoding module, (ii) sequence to sequence translation module, and (iii) content flow module.

Image Encoding Module: We represent the segmented image as a color-embedded image. Each region of this segmented image is represented by a color corresponding to this region's label, as shown in Figure 4.10. Color-embedded image is encoded by extracting deep filter bank features (FC-CNN) [114] with a pre-trained network. To enhance the discriminating power of FC-CNN, we fine-tune the pre-trained network with the color-embedded images and their corresponding label images. We use VGG-16 [104] as a pre-trained model and attach a deconvolution module similar to FCN32 [26] while fine-tuning the network.

Sequence-to-Sequence Translation Module:

It is a neural network that directly models the conditional probability $p(y|x)$ of translating a source vector sequence, $x_1, x_2, \ldots, x_n$ to a target vector sequence, $y_1, y_2, \ldots, y_m$. This module consists of two components: (i) an encoder that computes a representation s for each source vector and (ii) a

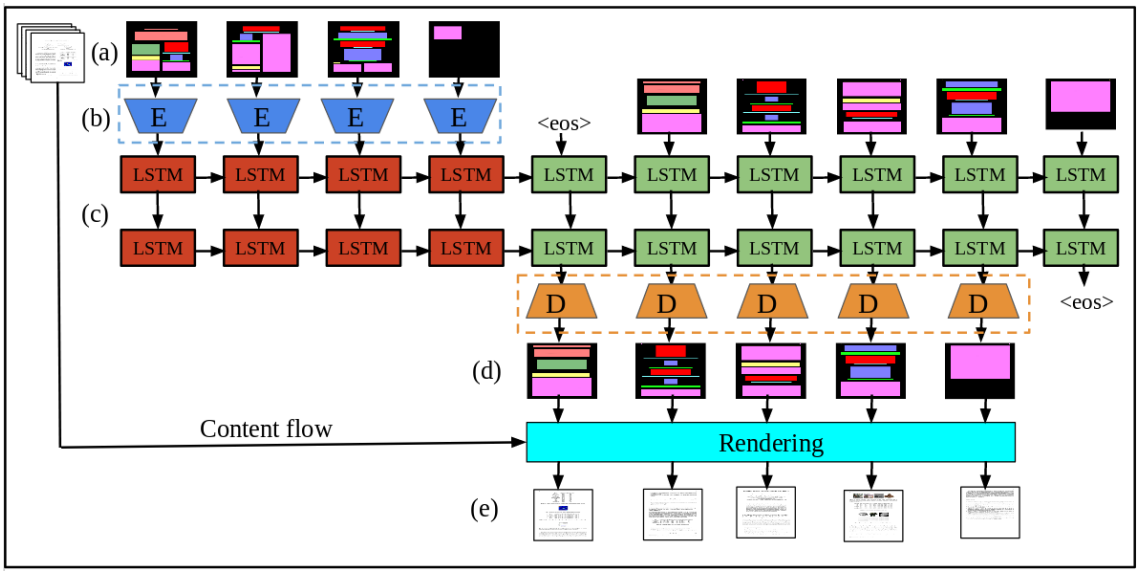


Figure 6.2 Overview of our document image reformatting module. (a) A sequence of segmented images corresponding to the source document. (b) An image encoder (E) encodes each of them into a vector. (c) A sequence-to-sequence translation model predicts the target sequence vector. (d) A decoder (D) decodes each of them into a segmented image. (e) Finally, the source document's content is rendered into predicted segmented images to obtain the target document.

decoder which generates one target vector at each time step. Therefore, we decompose the conditional probability as:

$$log[p(y|x)] = \sum_{j=0}^{m} log[p(y_j|y_{<j}, s_i)]. \tag{6.1}$$

The source vector sequence $x_1, x_2, \ldots, x_n$ corresponds to color-embedded images which are generated by the image encoding module. We use long short-term memory LSTM [320] as an encoder.

The decoder consists of an attention layer similar to [321], LSTM, and a convolution layer. The attention module is employed to reduce the burden of decoding the entire output sequence from a single encoded vector s. LSTM decodes the encoded vector s and predicts the output vector sequence $y_1, y_2, \ldots, y_m$. Each predicted vector is forwarded to a convolution layer to get an output as an image.

Content Flow Module:

Finally, the target document is created by rendering the source document's content into a predicted color-embedded image. We consider Tesseract [322], an optical character recognition (OCR) text extraction tool to extract text from semantic regions like title, abstract, section heading, paragraph, caption, etc. of the source document. We crop tables and figures from the source document using their semantic labels.

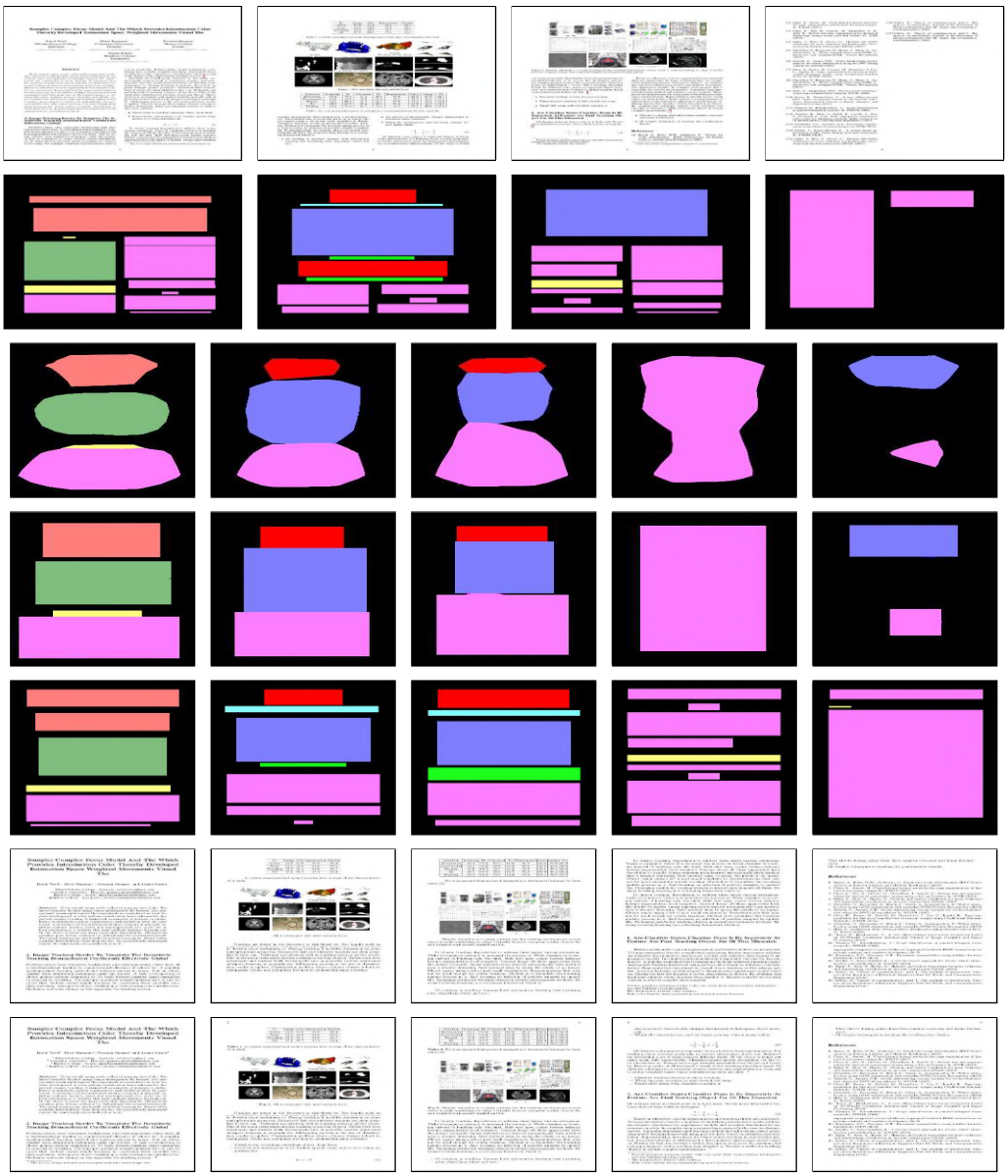


Figure 6.3 Reformatting result of CVPR style document images into ECCV style document images. **First row:** original CVPR formatted document images, **Second row:** corresponding layouts of these images, **Third row:** Predicted layout of the **eccv** style, **Fourth row:** predicted layouts after post-processing, **Fifth row:** ground truth layouts of ECCV style document images, **Sixth row:** reformatted document images of the ECCV style, **Seventh row:** original ECCV style document images.

We render tables, figures, and text information of each of the semantic regions (of the source document) into the corresponding predicted (i.e., target) segmented region. ImageMagic[1] is used for text rendering. Finally, we get a target document.

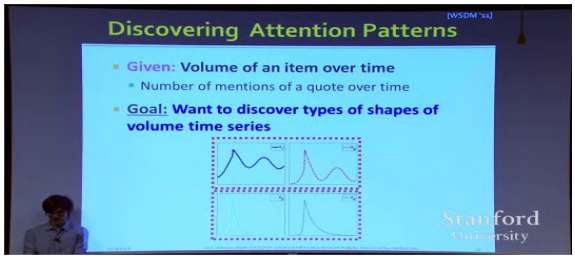
6.2 Classroom Slide Narration System

Slide presentations play an essential role in classroom teaching, technical lectures, massive open online courses (MOOC), and many formal meetings. Presentations are an efficient and effective tool for the students to understand the subject better. However, the visually impaired (VI) students do not benefit from this type of delivery due to their limitations in reading the slides. Human assistants could help VI students access and interpret the slide's content. However, it is impossible and impractical for the VI students to have a human reader every time because of the cost and the limited availability of trained personnel. From this perspective, this would be a primary issue in modern classrooms.

The automatic extraction of logical regions from these slide images is the initial step for understanding the slide's content. This can be defined as the segmentation of semantically similar regions, closely related to natural image semantic segmentation in computer vision [26, 29, 238]. The goal is to assign logical labels like title, paragraph, list, equation, table, and figure to each pixel in the slide images. Extracting meaningful regions become a challenging task due to the large variability in theme (i.e., the layout), style, and slide content. The text extraction from these regions is beneficial for the VI student only if the text is tagged with its logical function. The left image of Fig. 6.4, the plain text "*Discovering Attention Patterns*" is more meaningful to the VI student only if he knows the text is the heading of the slide. Similarly, detecting figures, equations, and table regions is also essential for understanding the slides. Our goal is to develop a system meaningfully describing the classroom slides with a markup text, as shown in Fig. 6.4.

The classroom slides have a complex design. The exact text could appear in various labels, such as "heading", "paragraph", and "list". Hence, only the visual features learned by the existing semantic segmentation networks (e.g., FCN [26], Deeplab [29], and PSPNet [238]) are not sufficient to distinguish various logical regions. Limited works [7, 8] on classroom slide segmentation utilize $(x,\ y)$ coordinate values to enhance the quality of segmentation results. However, none of these works [7, 8] utilize location information efficiently for segmentation. In contrast, Choi *et al.* [9] utilize the position encoding [30] to impose the height information in HANet for semantic segmentation of urban-scene images. In the case of slide images, the height and width information of each logical region are equally crucial for accurately segmenting regions. In this work, we propose a Classroom Slide Segmentation Network, called CSSNet to segment slide images accurately. The proposed network consists of (i) Attention Module, which utilizes the location encoding of the logical region (height and width of a region), and (ii) Atrous Spatial Pyramid Pooling (ASPP) module, which extracts multi-scale contextual features. The experiments on publicly available benchmark datasets WiSe [7] and SpaSe [8] establish the effective-

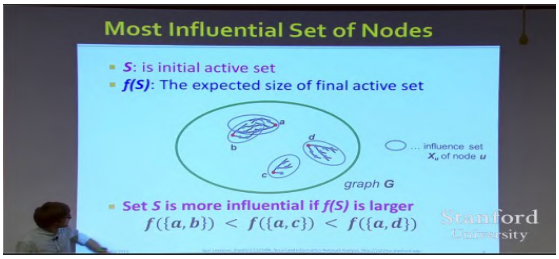
[1] `http://www.imagemagick.org`



Alt text: A screen shot of a person

Tesseract OCR: Volume of an item over time tions of a quote over time

Ours: <heading> Discovering Attention Patterns </heading> <list> = Given: Volume of an item over time Number of mentions of a quote over time </list> <list> Goal: Want to discover types of shapes of volume time series </list> <figure>Line graph</figure>



Alt text: A screenshot of a cell phone

Tesseract OCR:

Ours: <heading> Most Influential Set of Nodes </heading> <list> S is initial active set f of s: The expected size of final active set </list> <list> Set S is more influential if f(S) is larger </list> <Figure> Venn Diagram </Figure> <Figuretext> O ... influence set X, of node u </Figuretext><equation> f(a,b}) < f(a,c}) < f(a,d)</equation>

Figure 6.4 Illustrate qualitative comparison between outputs obtained by the proposed CSNS and the existing assistive systems — Automatic Alt-Text (AAT) [258] and Tesseract OCR [322]. The proposed system generates a markup text, where each logical content is tagged with its appropriate logical function. The audio file generated using this markup text gives a clear picture of the classroom slide's content to the blind or VI students.

ness of the proposed LEANet over state-of-the-art techniques — Haurilet *et al.* [7], Haurilet *et al.* [8], HANet [9], DANet [10], and DRANet [11].

The proposed Classroom Slide Narration System (CSNS) creates audio content to help VI students to read and understand the slide's content. The audio created corresponds to the segmented region's content in reading order. We use four existing recognizers —OCR [322], equation descriptor [5], table recognizer [323], and figure classifier [46] to recognize content from the segmented regions. We evaluate the effectiveness of the developed system on the WiSe dataset [7] and compare the user experiences with the existing assistive systems AAT [258] and Tesseract OCR [322]. The proposed system attains a better user experience than the existing assistive systems.

The contributions of this work are as follows:

- Proposes a CSSNet to efficiently segment the classroom slide images by utilizing the location encoding.

- Presents a CSNS as a use case of slide segmentation task to narrate the slides content in reading order. It helps the VI students read and understand the slides content.

- Perform experiments on publicly available benchmark datasets WiSe and SPaSe to establish the effectiveness of the proposed CSSNet and CSNS over existing slide segmentation and assistive techniques.

6.2.1 Classroom Slide Narration System

We developed a CSNS as a use case of slide segmentation to help VI students read and grasp the slide's content without human assistance. The proposed CSNS consists of three modules — (i) **Slide Segmentation Module:** to identify logical regions present in the slide, (ii) **Information Extraction Module:** to extract meaningful information from each segmented region, and (iii) **Audio Creation Module:** to create an audio narration of extracted information in proper order. We discuss each of these modules in detail.

Information Extraction Module

We coarsely group the classroom slide image into four logical regions — text, figure, equation, and table and use existing recognizers to recognize the content of the regions.

OCR: We use well-known Optical Character Recognition (OCR), Tesseract [322] to recognize the content of the text regions is presented in slide images.

Figure Classification: Figures mainly different types of plots, sketches, block diagrams,maps, and others. appear in slides. We use `DocFigure` [46] to find the category of each figure in slide images.

Equation Description: The equation frequently occurs in the classroom slides. We follow MED model [5] for describing equations in natural language to interpret the logical meaning of equations for VI students.

Table Structure Recognition: Recognition of the table present in the slide includes detecting cells, finding an association between cells, and recognizing the cell's content. We use TabStruct-Net [323] to recognize the physical structure of the table and then Tesseract [322] to recognize content.

The information from the four modules is combined and saved as a JSON file. These processes are done on the server, and then the JSON file is sent to the mobile app, and the VI student understands the slide's content by interacting with the app. **Audio Creation Module**

Based on a study that empirically investigates the difficulties experienced by VI internet users [324], a mobile application was developed for VI students to read and understand the content of the classroom slides. For this, we placed a camera in the classroom connected to the server to capture slide images. The VI students use this mobile app to connect to the server through a WiFi connection. Using the mobile application (**CAPTURE** button), the student can request the server to get information on the current slide. The server captures the current slide image through the camera, extracts the information as a JSON file, and sends it back to the mobile application. Fig. 6.5 shows a screenshot of the developed mobile app with one example. The app has two modes — (i) interactive and (ii) non-interactive. In the interactive mode, the user can navigate various logical regions of slides by themselves. The app shows the captured slide image with all extracted information. When the user touches one particular area, it automatically plays the audio sound using the android TTS library of that region's content along with the category label. In the non-interactive mode, the app automatically plays the audio corresponding to the slide's content. In this case, the screen contains the **READ ALL** button; when the user touches this

Figure 6.5 The user interface of the developed mobile application. The mobile screen displays the camera-captured slide image. Various colors highlight the slide's logical regions for better visualization. The READ ALL button plays generated audio sound corresponding to the slide's content.

particular area, the app automatically plays the audio sound of the full slide's content. These features are demonstrated in the demo video[2]

6.2.2 System Evaluation

We establish the effectiveness of the developed slide narration system by comparing the performance with Facebook's Automatic Alt-Text (AAT) [258] and Tesseract OCR [322]. We use 300 test images of the WiSe dataset for evaluating the system. 30 volunteers who can read and write English are selected. Each volunteer takes 300 slide images and evaluates corresponding audio outputs generated by various techniques. The volunteer gives a grade (value in the range $[0,\ 10]$) to the generated audio files using various techniques and comparing them with the original slide content. We collect all these grades and plot them as a bar chart (shown in Fig. 6.6). The user gives the best grade for the audio file corresponding to the ground truth segmentation of the slide. However, the proposed CSNS also performs comparatively better than the existing techniques.

6.2.2.1 Lecture-Slides Retrieval Evaluation

We utilize the suggested framework by training it on train data with automatic annotations and then assess its performance using a baseline established on manually annotated test data. Our dataset comprises two queries for each slide image: in the training set, there is both a generated summary and a paraphrased version of it, while in the testing and validation sets, we have the manually annotated summary and a paraphrase generated using ChatGPT [311]. During both the training and testing phases, we randomly select a sentence from a query associated with a slide image. It is important to note that all

[2] https://youtu.be/PnPYrA8ykF0

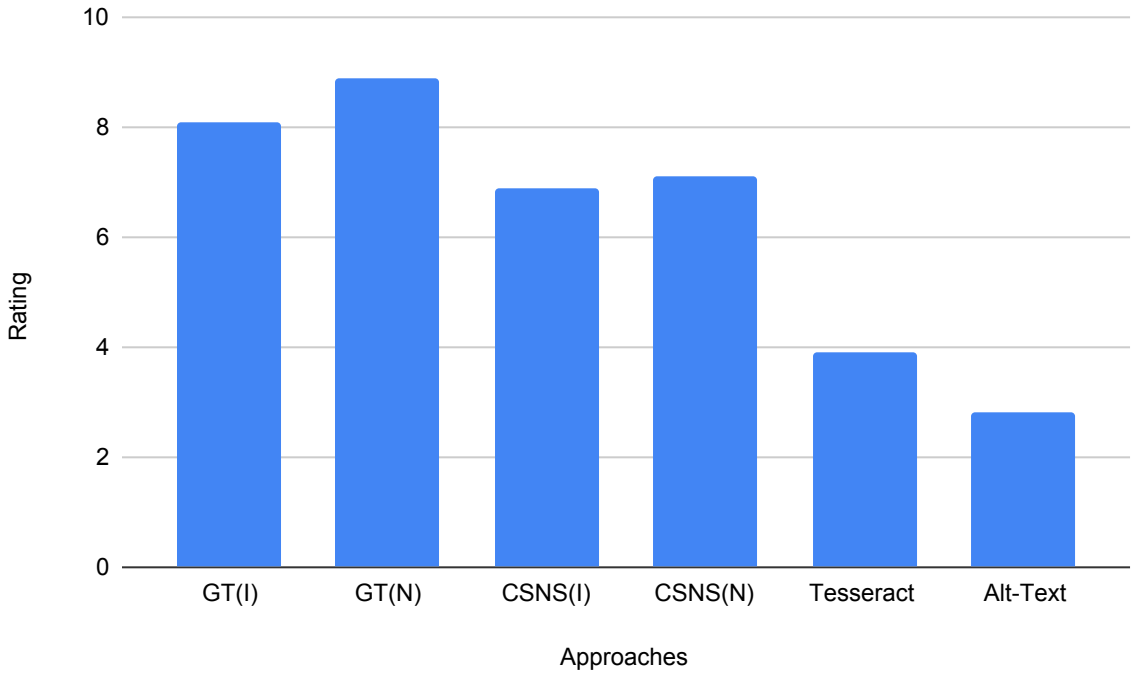


Figure 6.6 User rating on performance of various approaches. **I** and **N** indicate **interactive** and **non-interactive** modes, respectively.

results presented in this section are obtained by combining 50% of the manual summaries with 50% of their respective paraphrased sentences.

6.2.2.2 Implementation Details

In training, we use Adam optimizer with a base learning rate of 2^{-4} and weight decay of 10^{-2}. In the ViLT [325], we resize the shorter edge of input images to 384 and limit the longer edge to under 640 while preserving the aspect ratio. Patch projection of ViLT-B/32 yields $12 \times 20 = 240$ patches for an image with a resolution of $384\times$ 640. We interpolate V^{pos} of ViT-B/32 to fit the size of each image and pad the patches for batch training and the $\lambda_1, \lambda_2 \in [0.01, 0.001]$. To tokenize text inputs, we use the *bert-base-uncased* tokenizer and learn the textual embedding-related parameters t_{class} , T, and T^{pos} from scratch. We train the model for 225 K steps on 64 NVIDIA 4 GPUs with a batch size of 8.

6.3 Searching over a Large Collection of Lecture-Slides

Online education has become increasingly popular in recent years, partly due to its convenience and flexibility. As a result, there is now a greater demand for effective presentation materials to support this mode of learning. While plenty of lecture slide presentations on various topics are available online, searching for a relevant and compelling slide deck for a given query can be tedious and time-consuming for educators and students.

Further, in the retrieval task, using text input may not be the most suitable option in the following scenarios: i) when the user cannot recall the correct keyword for searching, but he/she has a picture in mind. ii) Students who have limited language proficiency struggle to express the search intent using text and may prefer to draw the concept. Towards accelerating research on this important topic, we present Lecture Slide Deck Search Engine [326](LecDeckSearch Engine) – a search engine that supports both natural language queries and hand-drawn sketches and performs a search on a very large-scale collection of slides. This search engine's functionalities are illustrated in Figure 6.7.

In recent years, researchers have been a growing interest in developing AI systems that use educational lecture videos and presentation slide images [7, 8, 55–58]. By using such a system, multiple AI systems can easily design new slides by combining search results and reusing figures and graphs from existing slide images, thereby minimizing manual effort. However, existing recommendation or retrieval systems [55,59] are designed to retrieve slides from video files and have several limitations. Firstly, they rely on the transcript in the video to retrieve slides, which are not always contextually aligned with the slide image and are also not always available, e.g., in lecture slide *image collection*. Secondly, these retrieval systems are restricted to text queries and do not support hand-drawn sketches of diagrams as queries. Finally, text queries often contain logical regions (semantic labels) like titles, bullet points, figures, etc., and figure types like line graphs, bar charts, Venn diagrams, tree diagrams, etc. The existing architectures are not explicitly designed to handle and learn these semantic regions and figure types. We propose the LecDeckSearch Engine Engine to overcome these limitations.

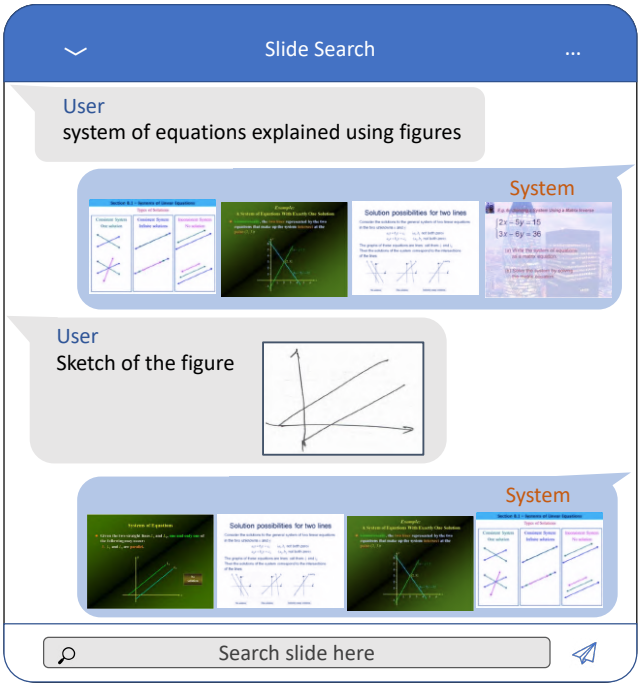


Figure 6.7 We present LecDeckSearch Engine – a semantic labels-aware transformer model that enables users to query and retrieve relevant lecture slides from a large collection using natural language summaries or hand-drawn sketches as queries. (Best viewed in color).

The LecDeckSearch Engine is a unique architecture that handles text and sketches and combines both queries. Compared to other slide retrieval approaches [55, 59], our model utilizes the semantic labels of slide images in the transformer model, where the text and image modalities of slide images are combined using a vision and language transformer (ViLT), and the queries are encoded with PIE-Net proposed in [275]. Our model demonstrates impressive performance on the lecture retrieval task and clearly outperforms related baselines. **Contributions:** We make the following contributions: (i) We present the LecSD – a very large-scale dataset of lecture slide images harvested from the web and associated annotations. The dataset includes around $50K$ slide images covering topics of *Data Structures* and manually written natural language summaries as well as hand-drawn sketch queries for searching figures. (ii) We propose a novel model that leverages the semantic label of slides and encodes them into a novel semantic label-aware transformer model. The representation learned using this model is used to score against the representations for natural language summary query or hand-drawn sketches to learn the relevance of slides given query. (iii) We perform extensive experiments and ablation to verify the efficacy and the limitations of our model. Our proposed approach significantly outperforms competitive approaches , establishing a new state-of-the-art for the task.

6.3.1 Semantic Labels-Aware Retrieval model for Lecture-Slides

Our goal is to retrieve the most appropriate slide image from the dataset given a query. Let $\mathcal{K} = \{(a_j, b_j)\}_{j=1}^{N}$ be the dataset consisting of a description of slide a_j and the slide image b_j. The description $a_j = (t, s)$ combines text description t and the sketch description s. Hence, the system supports both natural language and hand-drawn sketch queries. The goal is to learn an embedding space that can quantify the similarity between the slide image and description. As a result, given a description (text or sketch, or both) a_j, one could retrieve its similar slide images from $\{b_1, b_2, \ldots, b_N\}$.

We propose a novel slide image retrieval system as shown in Figure 6.8. We first describe the layout, figure type, and text extraction from lecture slide images in Section 6.3.1.1). The lecture slide encoder that encodes the layouts, figures, and texts along with the slide image is described in Section 6.3.1.2. The query text and query sketch encoder are explained in Section 6.3.1.3, and finally train the model with Multiple Instance Learning (MIL) [327] framework (Section 6.3.1.4), and provide the inference details.

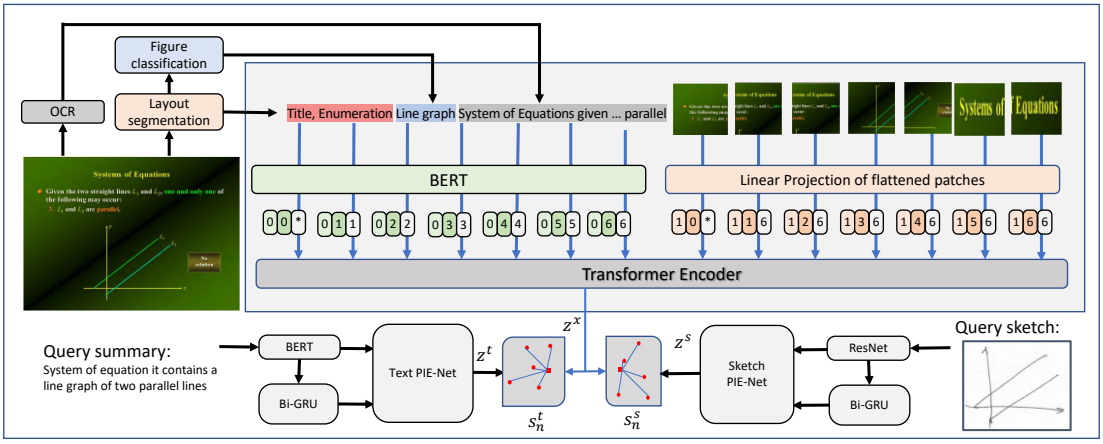


Figure 6.8 The proposed LecDeckSearch Engine architecture. The LecSD dataset is indexed by encoding features such as layout regions, figure classes, OCR text, slide image, and focus area using a ViL transformer. The query text and the sketch are independently encoded using PIE-Net [275]. The architecture predicts the final retrieval result based on the similarity score s_n^t and s_n^s. (best view in color)

6.3.1.1 Semantic Labelling of Lecture Slides

The query to retrieve a slide image need not contain all the text on the slide image. The query contains the keywords and logical regions of the slide image, such as title, enumeration, table, figures, and its various types. For example, "*system of equations explained using enumeration and a line graph*." In order to handle these queries, we propose a novel slide image retrieval system, as shown in Figure 6.8. Our approach utilizes a pre-trained slide image segmentation module [56] and a multi-feature

head model [313] trained on DocFig [46] dataset to obtain the logical regions t_l and the type of the figure present on the slide image t_g, respectively. In addition to these modules, we also use an Optical Character Recognizer (OCR) to extract the text t_o on the slide image.

6.3.1.2 Lecture Slide Image Data Encoder

In the dataset indexing, we collect the t_l, t_g, and t_o information along with the slide image b and a focus area b_f from each slide image. The focus area of a slide image is the possible logical region where the keyword is occurring. Most of the query keywords from our study are from the following logical regions: title, enumeration, paragraphs, and captions. In our proposed architecture, we chose the title area as a focus area. We chose the enumeration region if the title region was absent on the slide.

We encode the text and images using a Vision and Language Transformer (ViLT) [325]. The input text t is the concatenation of t_l, t_g, and t_o and $t \in \mathbb{R}^{L\times|V|}$ is embedded to $\bar{t} \in \mathbb{R}^{L\times H}$ with a word embedding matrix $T \in \mathbb{R}^{|V|\times H}$ and a position embedding matrix $T^{\text{pos}} \in \mathbb{R}^{(L+1)\times H}$. The input slide image and focus area concatenate to $bb = [b, b_f]$ and $bb \in \mathbb{R}^{C\times H\times W}$ is sliced into patches and flattened to $v \in \mathbb{R}^{N\times\left(P^2\cdot C\right)}$ where (P, P) is the patch resolution and $N = HW/P^2$. Followed by linear projection $V \in \mathbb{R}^{\left(P^2\cdot C\right)\times H}$ and position embedding $V^{\text{pos}} \in \mathbb{R}^{(N+1)\times H}$, v is embedded into $\bar{v} \in \mathbb{R}^{N\times H}$.

$$t = [t_l; t_g; t_o]\,. \tag{6.2}$$

$$\bar{t} = [t_{\text{class}}\,; t_1 T; \cdots; t_L T] + T^{\text{pos}}. \tag{6.3}$$

$$\bar{v} = [v_{\text{class}}\,; v_1 T; \cdots; v_L T] + V^{\text{pos}}. \tag{6.4}$$

$$z^0 = \left[\bar{t} + t^{\text{type}}; \bar{v} + v^{\text{type}}\right]. \tag{6.5}$$

$$\hat{z}^d = \text{MSA}\left(\text{LN}\left(z^{d-1}\right)\right) + z^{d-1}, \quad d = 1, \ldots, D. \tag{6.6}$$

$$z^d = \text{MLP}\left(\text{LN}\left(\hat{z}^d\right)\right) + \hat{z}^d, \quad d = 1, \ldots, D. \tag{6.7}$$

$$z^x = \tanh\left(z_0^D W_{\text{pool}}\right). \tag{6.8}$$

The embeddings are summed with their corresponding modal-type embedding vectors $t^{\text{type}}, v^{\text{type}} \in \mathbb{R}^H$, then, are concatenated into a combined sequence z^0. The z^0 vector is passed through the ViLT consisting of layer normalization (LN) stacked blocks, including a multiheaded self-attention (MSA) layer and an MLP layer. The contextualized vector z is iteratively updated through D depth transformer layers up until the final contextualized sequence z^d. Further, z^x is a pooled representation of the whole multimodal input. It is obtained by applying linear projection $W_{\text{pool}} \in \mathbb{R}^{H\times H}$ and hyperbolic tangent upon the first index of sequence z^D.

6.3.1.3 Query Feature Extraction

In our experiment, the query can be either a slide summary, sketch image, or a combination of both. The words in the text query t are encoded using pre-trained BERT model [219] and used as local features $\Psi(t) \in \mathbb{R}^{L\times 300}$ where L is the number of words in t. Then, we feed the local feature to a bi-GRU with H hidden units and take the final hidden states as global features $\phi(t) \in \mathbb{R}^H$. The query sketch image s is encoded using ResNet-152 [106]. The feature map before the final average pooling layer as local features $\Psi(s) \in \mathbb{R}^{7\times 7\times 2048}$. Further, we apply average pooling and feed the output to one fully connected layer to obtain global features $\phi(s) \in \mathbb{R}^H$.

The PIE-net proposed in [275] encodes the local and global features for text and sketch queries.

$$z^t = \text{textPIE-Net}(\Psi(t), \phi(t)) \tag{6.9}$$

$$z^s = \text{sketchPIE-Net}(\Psi(s), \phi(s)) \tag{6.10}$$

6.3.1.4 Optimization and Inference

We optimize our model to minimize the following loss function:

$$\mathcal{L} = \mathcal{L}_{mil} + \lambda_1 \mathcal{L}_{mmd} + \lambda_2 \mathcal{L}_{div}. \tag{6.11}$$

Where λ_1 and λ_2 are the scalar weights. $\mathcal{L}_{mil}$ is the multi-instance learning (MIL) loss [327] with the learning constraint for retrieval tasks.

The loss function $\mathcal{L}_{mil}$ only considers the minimum distance pair in the loss computation. Hence, the distribution induced by features may diverge quickly. Maximum Mean Discrepancy (MMD) [328] based loss $\mathcal{L}_{mmd}$ is introduced to regularize the discrepancy between the two distributions. The $\mathcal{L}_{div}$ is the diversity loss to ensure that the PIE-Net produces diverse representations of an instance. We follow these loss calculations described in [275].

In the training slide images of LecSD dataset, all the slide images do not contain figures. Hence, we first train the ViLT and the text PIE-Net models. Finally, the sketch PIE-Net models learn during the fine-tuning of the whole network with slide images having both summary and sketch queries.

In the inference stage, we assume that the dataset contains $\mathcal{N}$ lecture slide images. Further, the ViLT encoded vectors for i^{th} slide are represented as z_i^x. Now, given a query instance of slide summary and sketch image, we calculate an embedding vector z^t and z^s, respectively. Then, the similarity between the query and $i = 1^{st}$ to $\mathcal{N}^{th}$ lecture slides are computed as follows:

$$s_n^t = [sim(z^t, z_1^x), \cdots, sim(z^t, z_{\mathcal{N}}^x)], \tag{6.12}$$

$$s_n^s = [sim(z^s, z_1^x), \cdots, sim(z^s, z_{\mathcal{N}}^x)], \tag{6.13}$$

$$s_n = \frac{1}{2}(s_n^t + s_n^s). \tag{6.14}$$

Features	Summary to Slide			
	@1	@5	@10	Median
t_o, b	16.24	35.31	44.05	17.00
t_o, $[b; b_f]$	18.27	38.07	46.74	14.00
$[t_l; t_g; t_o]$, b	20.70	41.47	49.94	11.00
$[t_l; t_g]$, $[b; b_f]$	19.89	40.45	49.00	11.00
$[t_l; t_g; t_o]$, $[b; b_f]$	21.23	42.07	50.58	10.00

Table 6.1 Comparison study on the contribution of various features such as OCR text t_o, layout segmentation t_l, graphics type t_g, slide image b, and the focus area b_f on the retrieval task.

Here, s_n^t, s_n^s, and s_n are the similarity score of query text, sketch, and the combined respectively with the dataset of $\mathcal{N}$ instances. We then rank the database images with respect to these similarities.

6.3.2 Lecture Slide Image Retrieval using Natural Language Summary-based Query

To identify the contribution of various features, i.e., OCR text (t_o), layout region (t_l), graphics type (t_g), slide image (b), and the focus area (b_f) for the task, we conducted the ablation study, and the result is shown in Table 6.1. First, we used the t_o and i features in training and obtained 16.24% recall at one. Then added the i_f feature that improved a 2.03% improvement in the recall at one. Next, we combine $[t_l; t_g]$ with t_o, with b, which improves an addition of 2.5% improvement in the recall at one. This indicates that the layout features $[t_l; t_g]$ are efficient features for slide retrieval. Further, we removed the t_o, which resulted in a recall reduction of 0.81%. Finally, we used all the features and obtained a 21.23% recall at one.

We compare the proposed model on a summary-based slide image retrieval task, and the results are reported in Table 6.2. The proposed model outperforms the existing cross-model retrieval approaches. The PolyVilt [59] models inspire our model architecture. It uses the ViLT for image encoding. However, the proposed model outperforms the PolyVilt by 3.99% in the recall at one. The PVSE [275], PCME [274] models perform better in retrieving the natural images given a caption. However, these models are unable to learn logical regions from slide images. Hence, the performance was reduced, indicating that a layout segmentation module and document figure classification module are essential for slide image retrieval tasks.

We assessed the ability of the suggested model to generalize. We conducted this evaluation by testing the model, which was trained on slide images associated with the *data structure* topic, using two different sets of slide images related to *computer networks* and *optimization*. The results showed a significant performance decline of over 56%, indicating that all models, including the proposed one, struggled to retrieve slide images from topics that were previously unknown.

Figure 6.9 shows the qualitative result of the proposed approach. The system retrieves the slide image with the pseudo-code and block diagram in the first and second rows. We also show retrieval

Methods	*Data structure*			
	@1	@5	@10	Medien
Random	0.01	0.05	0.1	5000
PCME [274]	8.76	24.42	40.30	39.00
CLIP [273]	9.61	27.47	42.50	31.00
PVSE [275]	13.63	31.55	43.72	25.00
PolyViLT [59]	16.24	35.31	44.05	17.00
Ours	20.23	42.07	50.58	10.00

Table 6.2 The summary-based slide image retrieval performance of multi-model slide image retrieval models, which were trained on slide images from the *data structure* topic.

Methods	*Computer networks*				*Optimization*			
	@1	@5	@10	Medien	@1	@5	@10	Medien
Random	0.05	0.25	0.5	1000	0.05	0.25	0.5	1000
PCME [274]	6.65	18.64	27.56	56.00	6.21	14.32	22.21	173
CLIP [273]	-	-	-	-	-	-	-	-
PVSE [275]	7.5	19.43	25.84	57.00	6.32	14.54	22.54	165
PolyViLT [59]	7.94	21.53	27.75	53.00	6.75	16.45	23.75	150
Ours	8.87	22.01	30.33	41.00	7.51	17.55	24.18	132.00

Table 6.3 The summary-based slide image retrieval performance of different multi-model slide image retrieval models, which were trained on slide images from the *data structure* topic, was evaluated on slides related to the topics of *Data structure*, *computer networks*, and *optimization*.

results when a hand-drawn sketch of a diagram is used as a query in the last row. The ground truth for this query is highlighted in the green border.

6.3.3 Refining Lecture Slide Image Retrieval using Hand-drawn Sketch Query

Query type	Slide retrieval			
	@1	@5	@10	Median
Sketch	23.63	51.55	62.72	5.00
Summary	37.05	61.32	65.67	4.00
Combined	41.50	64.00	68.50	2.00

Table 6.4 Slide image retrieval result given only sketches, only text summary, and the combination of text and sketch on the slide images from the topic *data structure*

In testing, we infer the result based on text summary similarity s_n^t, sketch similarity s_n^s, and the combined similarity s_n; the result is shown in Table 6.4. Here, we noticed that the combined similarity between the sketch and the summary improved the retrieval result to 41.5%. Forth row in Figure 6.9

shows the sketch query and the successfully re-ranked result, which matches the sketch query. However, the sketch-based retrieval fails for the figure having a smaller size compared to the slide image size.

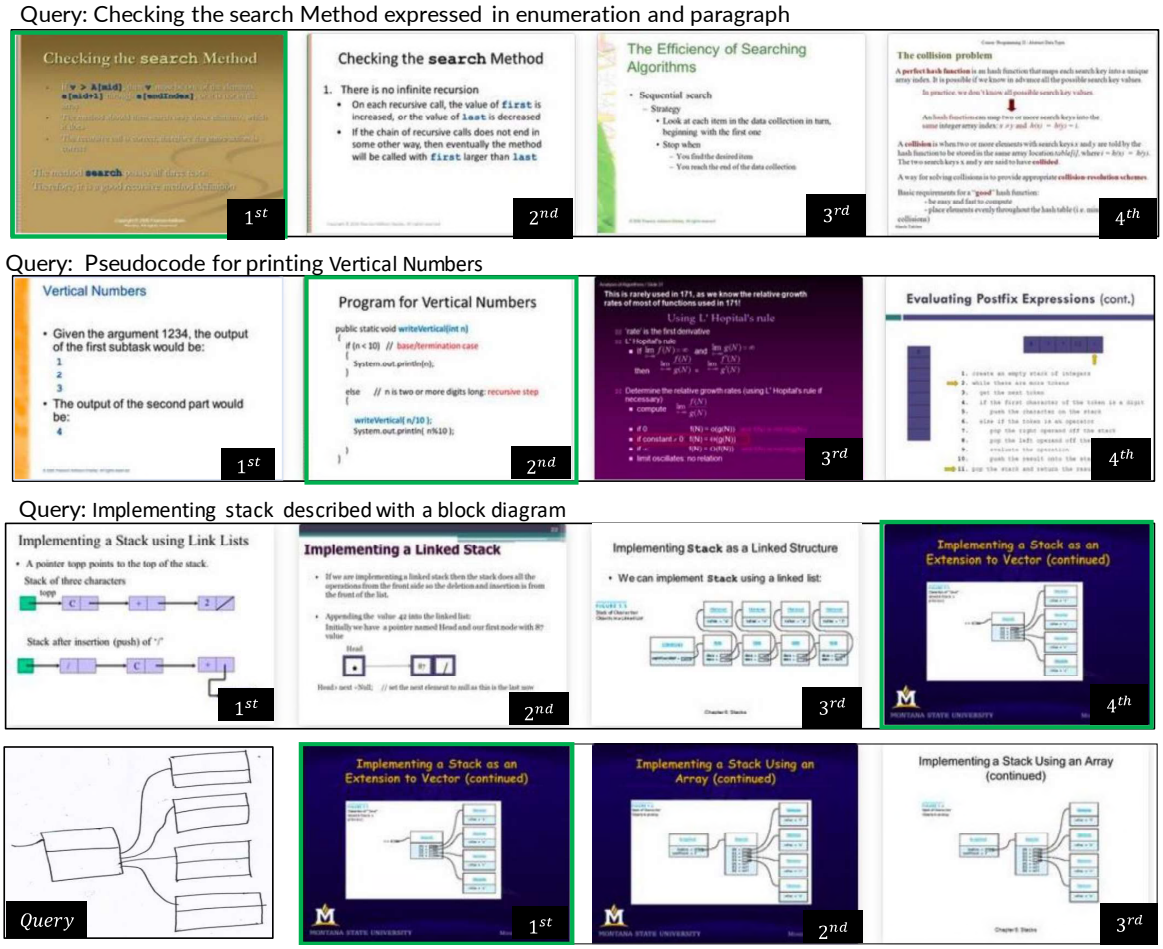


Figure 6.9 Qualitative result of proposed LecDeckSearch Engine. The text query and its result are shown in the top three rows. The last row shows the re-ranked result given the sketch query. The slide in the green bounding box indicates the correct image for the query. More qualitative analysis is provided in the supplementary material. (best viewed in color).

6.4 Discussion

This chapter delves into three innovative applications that leverage document image analysis and layout segmentation advancements. These applications, namely template transfer, slide narration system, and slide image retrieval, offer remarkable solutions to some pressing challenges in document management and education.

The evolution of document layout segmentation intricately links the performance of the algorithms underpinning these applications. As we have explored throughout this work, accurate document layout

segmentation is a pivotal step in understanding the spatial organization of a document. It provides the foundation for extracting meaningful content and contextual information from various documents. Consequently, they empower these applications as segmentation techniques advance by providing more structured and relevant data.

One of the notable applications that benefit from our work is the slide image retrieval system. This system could revolutionize education by enabling automatic slide creation based on human commands. It addresses the growing need for personalized learning experiences in the Massive Open Online Courses (MOOCs) era. By allowing users to retrieve and adapt existing slide content effortlessly, this system promotes interactive and adaptive learning, significantly enhancing the educational landscape.

In the subsequent sections of this chapter, we will delve into each of these applications, shedding light on their significance and the synergy between document layout segmentation and performance improvements. These applications exemplify how the continual refinement of document analysis techniques opens up exciting new possibilities in education and beyond.

Chapter 7

Conclusion and Future Work

In this thesis, we address the intricate challenges within document image analysis, offering innovative solutions, particularly in document figure classification, layout analysis, and retrieval. Additionally, we introduce two invaluable datasets: `DocFigure` and LecSD. As part of this research work, we have developed three practical applications in document image analysis: document reformatting, a classroom slide narration system, and a slide deck retrieval system. The comprehensive conclusion section provides insights into these contributions and outlines potential directions for future research.

7.1 Multi-modular Feature

We introduce a deep multi-modular feature extraction architecture for various classification tasks in document image analysis. The proposed architecture extracts three features - discriminative, encoded/texture, and global. Diverse experiments conclude that a combination of discriminative, texture, and global features performs better for various classification tasks - document image classification, genre classification of book, document figure classification, and script identification. We visualize the L_2 norm of the learned global and discriminative features in the form of the heat map, highlighting their importance for various classification tasks. The heat maps highlight that discriminative features are more useful for document image classification, document figure classification, and genre classification than other features. In contrast, the encoded feature is essential compared to the other features for the script identification task. In the future, we will explore the proposed architecture for classroom slide retrieval and signature verification.

7.2 Deep Feature based Document Image Segmentation

We have proposed a deep feature-based document image segmentation approach. This approach has the following advantages compared to other methods: (i) It provides better feature representation for document image regions. (ii) Since the proposed approach uses a pre-trained network, the training time is significantly reduced compared to an end-to-end CNN training approach [178] because the proposed

method uses SVM to train the model. As a future work, we will try the FCN [26] architecture which wil be more suites in document segmentation.

7.3 DocFigure Dataset

In this article, we introduced a dataset DocFigure with 33K figure images of 28 different categories present in the scientific articles. Our generated dataset is dedicated to the document figure classification task. Here, we also designed an efficient web-based annotation tool to annotate 33K images with minimum efforts of human annotators. We also proposed three baseline approaches based on deep feature (FC-CNN), deep texture feature (FV-CNN) and concatenation of both these features to validate our generated dataset on the document figure classification task. Experimentally, we concluded that the concatenation of the deep feature and deep texture feature is more effective for the figure classification task.

7.4 Classroom Slide Segmentation Network

This paper presents a classroom slide segmentation network to segment classroom slide images more accurately. The attention module in the proposed network enriches the learned contextual features by utilizing the location encoding of the logical region. We developed a Classroom Slide Narration System as an application of classroom slide segmentation for helping VI students read and understand the slide's content. Experiments on benchmark datasets WiSe and SpaSe conclude that the proposed LEANet achieves the best slide segmentation results as compared to the state-of-the-art methods. We also demonstrate the effectiveness of our developed Classroom Slide Narration System over existing assistive tools in terms of user experiences.

7.5 Document image reformatting

We propose a document image reformatting framework based on a sequence-to-sequence translation model. Segmentation of document images is the first step of our solution. For this purpose, we employ a document image segmentation model developed using PSPNet. We introduce two approaches for generating many document images with limited annotation errors on document image segmentation tasks due to the scarcity of annotated pixel-wise ground truth document images. Furthermore, we demonstrate the performance of our reformatting framework on reformation of CVPR style document images into target document images of ECCV style. Since the reformatting module is trainable and can learn the document's layout, it produces good results. However, the performance of the reformatting task depends on the accuracy of the segmentation task. The segmentation module performs well with the limited annotated training images. However, it sometimes fails to produce good segmentation output for regions (like figure caption, table caption, and figure with sparse content). For the reformatting task,

after transferring layouts from CVPR to ECCV, we render tables, figures, and text information of each of the semantic regions (of the source document) into the corresponding predicted (i.e., target) segmented region. In the future, we will concentrate on designing a network that will transfer the source document's layout and content to the target document automatically.

7.6 LecDeckSearch Engine

We introduced the LecSD - a comprehensive collection of lecture slide decks intended to serve as a benchmark for the development of educational AI systems. The dataset is unique and has rich annotations, and it was specifically created to tackle two challenging research tasks that are relevant to education: i) retrieval of lecture slide images based on brief descriptions that include logical regions and figure classes and ii) retrieval of lecture slide images with figures based on hand-drawn sketches of the figures. Our benchmarking efforts revealed that existing retrieval models fall short of accurately identifying logical regions and figure classes. On the contrary, we proposed a new retrieval model called LecDeckSearch Engine, which is semantic labels-aware and includes sketch-based retrieval functionality. Nonetheless, the LecDeckSearch Engine does have a few drawbacks. First, it relies on an off-the-shelf layout segmentation module. Second, it struggles to retrieve diagrams with limited areas compared to slide images when given a sketch. Lastly, the model is unsuccessful in finding slides from unfamiliar subjects. We leave this as the future scope of this paper.

Related Publications

[P1] **K. V. Jobin**, Anand Mishra, and C.V. Jawahar. "Semantic Labels-Aware Transformer Model for Searching over a Large Collection of Lecture-Slides", in IEEE/CVF Winter Conference on Applications of Computer Vision (WACV), 2024.

[P2] **K.V. Jobin**, Ajoy Mondal, and C. V. Jawahar. "Document Image Analysis Using Deep Multimodular Features." *SN Computer Science* 4.1 (2022): 5.

[P3] **K. V. Jobin**, Ajoy Mondal, and C. V. Jawahar. "Classroom Slide Narration System." *International Conference on Computer Vision and Image Processing*. Cham: Springer International Publishing, 2021.

[P4] **K. V. Jobin**, Ajoy Mondal, and C. V. Jawahar. "Docfigure: A dataset for scientific document figure classification." 2019 *International Conference on Document Analysis and Recognition Workshops (ICDARW).* Vol. 1. IEEE, 2019.

[P5] **K. V. Jobin**, and C. V. Jawahar. "Document image segmentation using deep features." *Computer Vision, Pattern Recognition, Image Processing, and Graphics*: 6th National Conference, NCVPRIPG 2017, Mandi, India, December 16-19, 2017, Revised Selected Papers 6. Springer Singapore, 2018.

Bibliography

[1] S. N. Srihari and E. J. Kuebert, "Integration of hand-written address interpretation technology into the united states postal service remote computer reader system," in *Proceedings of the Fourth International Conference on Document Analysis and Recognition*, vol. 2. IEEE, 1997, pp. 892–896.

[2] S. N. Srihari, V. Govindaraju, and A. Shekhawat, "Interpretation of handwritten addresses in us mailstream," in *Proceedings of 2nd International Conference on Document Analysis and Recognition (ICDAR'93)*. IEEE, 1993, pp. 291–294.

[3] M. Mathew, A. K. Singh, and C. Jawahar, "Multilingual ocr for indic scripts," in *2016 12th IAPR Workshop on Document Analysis Systems (DAS)*. IEEE, 2016, pp. 186–191.

[4] K. Dutta, P. Krishnan, M. Mathew, and C. Jawahar, "Improving cnn-rnn hybrid networks for handwriting recognition," in *2018 16th international conference on frontiers in handwriting recognition (ICFHR)*. IEEE, 2018, pp. 80–85.

[5] A. Mondal and C. V. Jawahar, "Textual description for mathematical equations," in *ICDAR*, 2019.

[6] J. Calvo-Zaragoza, J. H. Jr, and A. Pacha, "Understanding optical music recognition," *ACM Computing Surveys (CSUR)*, vol. 53, no. 4, pp. 1–35, 2020.

[7] M. Haurilet, A. Roitberg, M. Martinez, and R. Stiefelhagen, "WiSe - slide segmentation in the wild," in *ICDAR*, 2019.

[8] Z. A.-H. Monica Haurilet and R. Stiefelhagen, "Spase - multi-label page segmentation for presentation slides," in *WACV*, 2019.

[9] S. Choi, J. T. Kim, and J. Choo, "Cars can't fly up in the sky: Improving urban-scene segmentation via height-driven attention networks," in *CVPR*, 2020.

[10] J. Fu, J. Liu, H. Tian, Y. Li, Y. Bao, Z. Fang, and H. Lu, "Dual attention network for scene segmentation," in *CVPR*, 2019.

[11] J. Fu, J. Liu, J. Jiang, Y. Li, Y. Bao, and H. Lu, "Scene segmentation with dual relation-aware attention network," *IEEE Trans. on Neural Networks and Learning Systems*, 2020.

[12] L. O'Gorman, "The document spectrum for page layout analysis," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 15, no. 11, pp. 1162–1173, 1993.

[13] K. Kise, A. Sato, and M. Iwata, "Segmentation of page images using the area voronoi diagram," *Computer Vision and Image Understanding*, vol. 70, no. 3, pp. 370–382, 1998.

[14] F. M. Wahl, K. Y. Wong, and R. G. Casey, "Block segmentation and text extraction in mixed text/image documents," *Computer graphics and image processing*, vol. 20, no. 4, pp. 375–390, 1982.

[15] G. Nagy, S. Seth, and M. Viswanathan, "A prototype document image analysis system for technical journals," *Computer*, vol. 25, no. 7, pp. 10–22, 1992.

[16] R. W. Smith, "Hybrid page layout analysis via tab-stop detection," in *2009 10th International Conference on Document Analysis and Recognition*. IEEE, 2009, pp. 241–245.

[17] J. Bernsen, "Dynamic thresholding of grey-level images,[in:] proc. of the 8th int," in *Conf. on Pattern Recognition*, 1986.

[18] J. Sauvola and M. Pietikäinen, "Adaptive document image binarization," *Pattern recognition*, vol. 33, no. 2, pp. 225–236, 2000.

[19] W. Niblack, *An introduction to digital image processing*. Strandberg Publishing Company, 1985.

[20] N. Otsu *et al.*, "A threshold selection method from gray-level histograms," *Automatica*, vol. 11, no. 285-296, pp. 23–27, 1975.

[21] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 6, pp. 1137–1149, 2016.

[22] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector," in *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14*. Springer, 2016, pp. 21–37.

[23] M. Liao, B. Shi, X. Bai, X. Wang, and W. Liu, "Textboxes: A fast text detector with a single deep neural network," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 31, 2017.

[24] M. Minouei, M. R. Soheili, and D. Stricker, "Document layout analysis with an enhanced object detector," in *2021 5th International Conference on Pattern Recognition and Image Analysis (IPRIA)*. IEEE, 2021, pp. 1–5.

[25] J. Hosang, R. Benenson, and B. Schiele, "Learning non-maximum suppression," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4507–4515.

[26] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *CVPR*, 2015.

[27] M. Mehri, A. Sellami, and S. Tabbone, "Historical document image segmentation combining deep learning and gabor features," in *International Conference on Document Analysis and Recognition*. Springer, 2023, pp. 395–410.

[28] D. Deng, H. Liu, X. Li, and D. Cai, "Pixellink: Detecting scene text via instance segmentation," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 32, 2018.

[29] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs," *PAMI*, 2017.

[30] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," *arXiv preprint arXiv:1706.03762*, 2017.

[31] Y. Zhou and C. L. Tan, "Learning-based scientific chart recognition," in *IWGR*, 2001.

[32] W. Huang and C. L. Tan, "A system for understanding imaged infographics and its applications," in *ACM-SDE*, 2007.

[33] B. Tang, X. Liu, J. Lei, M. Song, D. Tao, S. Sun, and F. Dong, "Deepchart: Combining deep convolutional networks and deep belief networks in chart classification," *Signal Processing*, 2016.

[34] N. Siegel, Z. Horvitz, R. Levin, S. Divvala, and A. Farhadi, "Figureseer: Parsing result-figures in research papers," in *ECCV*, 2016.

[35] V. S. N. Prasad, B. Siddiquie, J. Golbeck, and L. S. Davis, "Classifying computer generated charts," in *CBMI*, 2007.

[36] V. Karthikeyani and S. Nagarajan, "Machine learning classification algorithms to recognize chart types in portable document format (pdf) files," *IJCA*, 2012.

[37] M. Savva, N. Kong, A. Chhajta, L. Fei-Fei, M. Agrawala, and J. Heer, "Revision: Automated classification, analysis and redesign of chart images," in *User interface software and technology*, 2011.

[38] Google, "Lookout - assisted vision – apps on google play (no date) google," https://play.google.com/store/apps/details?id=com.google.android.apps.accessibility.reveal&hl=en_IN.

[39] "Clarifai, the world's leading computer vision platform (no date) clarifai," https://www.clarifai.com/(Accessed:10September2024)..

[40] K. V. Jobin and C. V. Jawahar, "Document image segmentation using deep features," in *Computer Vision, Pattern Recognition, Image Processing, and Graphics: 6th National Conference, NCVPRIPG 2017, Mandi, India, December 16-19, 2017, Revised Selected Papers 6*. Springer, 2018, pp. 372–382.

[41] X. Zhong, J. Tang, and A. J. Yepes, "Publaynet: largest dataset ever for document layout analysis," in *ICDAR*, 2019.

[42] B. K. Iwana, S. T. R. Rizvi, S. Ahmed, A. Dengel, and S. Uchida, "Judging a book by its cover," *arXiv*, 2016.

[43] A. Antonacopoulos, D. Bridson, C. Papadopoulos, and S. Pletschacher, "A realistic dataset for performance evaluation of document layout analysis," in *ICDAR*, 2009.

[44] A. Das, S. Roy, U. Bhattacharya, and S. K. Parui, "Document image classification with intra-domain transfer learning and stacked generalization of deep convolutional neural networks," in *ICPR*, 2018.

[45] M. Z. Afzal, A. Kolsch, S. Ahmed, and M. Liwicki, "Cutting the error by half investigation of very deep cnn and advanced training strategies for document image classification," in *ICDAR*, 2017.

[46] K. Jobin, A. Mondal, and C. Jawahar, "Docfigure: A dataset for scientific document figure classification," in *2019 International Conference on Document Analysis and Recognition Workshops (ICDARW)*, vol. 1. IEEE, 2019, pp. 74–79.

[47] K. Ubul, G. Tursun, A. Aysa, D. Impedovo, G. Pirlo, and T. Yibulayin, "Script identification of multi-script documents: A survey," *IEEE Access*, 2017.

[48] K. Torkkola, "Discriminative features for text document classification," *Formal Pattern Analysis & Applications*, 2004.

[49] H. Jiang, Z. Pan, and P. Hu, "Discriminative learning of generative models: large margin multinomial mixture models for document classification," *Pattern Analysis and Applications*, 2015.

[50] H. Soleimani and D. J. Miller, "Exploiting the value of class labels on high-dimensional feature spaces: topic models for semi-supervised document classification," *Pattern Analysis and Applications*, 2019.

[51] A. K. Singh, A. Mishra, P. Dabral, and C. Jawahar, "A simple and effective solution for script identification in the wild," in *DASW*, 2016.

[52] B. Shi, X. Bai, and C. Yao, "Script identification in the wild via discriminative convolutional neural network," *Pattern Recognition*, 2016.

[53] Y. P. Zhou and C. L. Tan, "Hough technique for bar charts detection and recognition in document images," in *ICIP*, 2000.

[54] ——, "Bar charts recognition using hough based syntactic segmentation," in *ICTAD*, 2000.

[55] D. Singh, G. Anchit, C. V. Jawahar, and M. Tapaswi, "Unsupervised audio-visual lecture segmentation," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2023, pp. 5232–5241.

[56] K. V. Jobin, A. Mondal, and C. V. Jawahar, "Classroom slide narration system," in *International Conference on Computer Vision and Image Processing*. Springer, 2021, pp. 135–146.

[57] D. Mahapatra, R. Mariappan, V. Rajan, K. Yadav, and S. Roy, "Videoken: Automatic video summarization and course curation to support learning," in *Companion Proceedings of the The Web Conference 2018*, 2018, pp. 239–242.

[58] D. Griol and Z. Callejas, "An architecture to develop multimodal educative applications with chatbots," *International Journal of Advanced Robotic Systems*, vol. 10, no. 3, p. 175, 2013.

[59] D. W. Lee, C. Ahuja, P. P. Liang, S. Natu, and L.-P. Morency, "Multimodal lecture presentations dataset: Understanding multimodality in educational slides," *arXiv preprint arXiv:2208.08080*, 2022.

[60] C. Harris, M. Stephens *et al.*, "A combined corner and edge detector," in *Alvey vision conference*, vol. 15. Citeseer, 1988, pp. 10–5244.

[61] J. Shi *et al.*, "Good features to track," in *1994 Proceedings of IEEE conference on computer vision and pattern recognition*. IEEE, 1994, pp. 593–600.

[62] E. Rosten and T. Drummond, "Fusing points and lines for high performance tracking," in *Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1*, vol. 2. Ieee, 2005, pp. 1508–1515.

[63] A. Semma, Y. Hannad, I. Siddiqi, C. Djeddi, and M. E. Y. El Kettani, "Writer identification using deep learning with fast keypoints and harris corner detector," *Expert Systems with Applications*, vol. 184, p. 115473, 2021.

[64] A. Gari, G. Khaissidi, M. Mrabti, D. Chenouni, and M. El Yacoubi, "Skew detection and correction based on hough transform and harris corners," in *2017 International Conference on Wireless Technologies, Embedded and Intelligent Systems (WITS)*. IEEE, 2017, pp. 1–4.

[65] F. Nourbakhsh, P. B. Pati, and A. Ramakrishnan, "Document page layout analysis using harris corner points," in *2006 Fourth International Conference on Intelligent Sensing and Information Processing*. IEEE, 2006, pp. 149–152.

[66] N. D. Modi, C. N. Paunwala, C. K. Modi, and S. Patnaik, "Skew correction for vehicle license plates using principal component of harris corner feature," in *2011 International Conference on Communication Systems and Network Technologies*. IEEE, 2011, pp. 339–343.

[67] D. G. Lowe, "Object recognition from local scale-invariant features," in *ICCV*, vol. 2, 1999.

[68] H. Bay, T. Tuytelaars, and L. Van Gool, "Surf: Speeded up robust features," in *Computer Vision–ECCV 2006: 9th European Conference on Computer Vision, Graz, Austria, May 7-13, 2006. Proceedings, Part I 9*. Springer, 2006, pp. 404–417.

[69] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "Orb: An efficient alternative to sift or surf," in *2011 International conference on computer vision*. Ieee, 2011, pp. 2564–2571.

[70] M. Calonder, V. Lepetit, C. Strecha, and P. Fua, "Brief: Binary robust independent elementary features," in *Computer Vision–ECCV 2010: 11th European Conference on Computer Vision, Heraklion, Crete, Greece, September 5-11, 2010, Proceedings, Part IV 11*. Springer, 2010, pp. 778–792.

[71] G. Rajput and S. B. Ummapure, "Script identification from handwritten documents using sift method," in *2017 IEEE international conference on power, control, signals and instrumentation engineering (ICPCSI)*. IEEE, 2017, pp. 520–526.

[72] M. Zahedi and S. Eslami, "Farsi/arabic optical font recognition using sift features," *Procedia Computer Science*, vol. 3, pp. 1055–1059, 2011.

[73] S. Ahmed, M. Liwicki, and A. Dengel, "Extraction of text touching graphics using surf," in *2012 10th IAPR International Workshop on Document Analysis Systems*. IEEE, 2012, pp. 349–353.

[74] M. Kumar, S. Gupta, and N. Mohan, "A computational approach for printed document forensics using surf and orb features," *Soft Computing*, vol. 24, pp. 13 197–13 208, 2020.

[75] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, vol. 1. Ieee, 2005, pp. 886–893.

[76] T. Ojala, M. Pietikainen, and T. Maenpaa, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 24, no. 7, pp. 971–987, 2002.

[77] F. Cruz, N. Sidere, M. Coustaty, V. P. d'Andecy, and J.-M. Ogier, "Local binary patterns for document forgery detection," in *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, vol. 1. IEEE, 2017, pp. 1223–1228.

[78] A. Nicolaou, M. Liwicki, and R. Ingold, "Oriented local binary patterns for writer identification." *AFHA*, vol. 1022, pp. 15–20, 2013.

[79] S. Dey, A. Nicolaou, J. Llados, and U. Pal, "Local binary pattern for word spotting in handwritten historical document," in *Structural, Syntactic, and Statistical Pattern Recognition: Joint IAPR International Workshop, S+ SSPR 2016, Mérida, Mexico, November 29-December 2, 2016, Proceedings*. Springer, 2016, pp. 574–583.

[80] S. Tian, S. Lu, B. Su, and C. L. Tan, "Scene text recognition using co-occurrence of histogram of oriented gradients," in *2013 12th International Conference on Document Analysis and Recognition*. IEEE, 2013, pp. 912–916.

[81] S. Tian, U. Bhattacharya, S. Lu, B. Su, Q. Wang, X. Wei, Y. Lu, and C. L. Tan, "Multilingual scene character recognition with co-occurrence of histogram of oriented gradients," *Pattern Recognition*, vol. 51, pp. 125–134, 2016.

[82] Sivic and Zisserman, "Video google: A text retrieval approach to object matching in videos," in *Proceedings ninth IEEE international conference on computer vision*. IEEE, 2003, pp. 1470–1477.

[83] G. Csurka, C. Dance, L. Fan, J. Willamowski, and C. Bray, "Visual categorization with bags of keypoints," in *Workshop on statistical learning in computer vision, ECCV*, vol. 1. Prague, 2004, pp. 1–2.

[84] S. Singh, A. Gupta, and A. A. Efros, "Unsupervised discovery of mid-level discriminative patches," in *Computer Vision–ECCV 2012: 12th European Conference on Computer Vision, Florence, Italy, October 7-13, 2012, Proceedings, Part II 12*. Springer, 2012, pp. 73–86.

[85] M. Juneja, A. Vedaldi, C. Jawahar, and A. Zisserman, "Blocks that shout: Distinctive parts for scene classification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 923–930.

[86] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *IJCV*, 2004.

[87] A. Bosch, A. Zisserman, and X. Munoz, "Scene classification via plsa," in *Computer Vision–ECCV 2006: 9th European Conference on Computer Vision, Graz, Austria, May 7-13, 2006, Proceedings, Part IV 9*. Springer, 2006, pp. 517–530.

[88] L. Fei-Fei and P. Perona, "A bayesian hierarchical model for learning natural scene categories," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 2. IEEE, 2005, pp. 524–531.

[89] D. Nister and H. Stewenius, "Scalable recognition with a vocabulary tree," in *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, vol. 2. Ieee, 2006, pp. 2161–2168.

[90] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman, "Object retrieval with large vocabularies and fast spatial matching," in *2007 IEEE conference on computer vision and pattern recognition*. IEEE, 2007, pp. 1–8.

[91] S. Lazebnik, C. Schmid, and J. Ponce, "Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories," in *2006 IEEE computer society conference on computer vision and pattern recognition (CVPR'06)*, vol. 2. IEEE, 2006, pp. 2169–2178.

[92] H. Jégou, M. Douze, C. Schmid, and P. Pérez, "Aggregating local descriptors into a compact image representation," in *CVPR*, 2010.

[93] F. Perronnin and C. Dance, "Fisher kernels on visual vocabularies for image categorization," in *2007 IEEE conference on computer vision and pattern recognition*. IEEE, 2007, pp. 1–8.

[94] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner *et al.*, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, 1998.

[95] A. Vedaldi and K. Lenc, "Matconvnet: Convolutional neural networks for matlab," in *Proceedings of the 23rd ACM international conference on Multimedia*, 2015, pp. 689–692.

[96] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proceedings of the 27th international conference on machine learning (ICML-10)*, 2010, pp. 807–814.

[97] I. Goodfellow, D. Warde-Farley, M. Mirza, A. Courville, and Y. Bengio, "Maxout networks," in *International conference on machine learning*. PMLR, 2013, pp. 1319–1327.

[98] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1026–1034.

[99] D.-A. Clevert, T. Unterthiner, and S. Hochreiter, "Fast and accurate deep network learning by exponential linear units (elus)," *arXiv preprint arXiv:1511.07289*, 2015.

[100] P. Ramachandran, B. Zoph, and Q. V. Le, "Searching for activation functions," *arXiv preprint arXiv:1710.05941*, 2017.

[101] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*. JMLR Workshop and Conference Proceedings, 2010, pp. 249–256.

[102] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *nature*, vol. 323, no. 6088, pp. 533–536, 1986.

[103] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012.

[104] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv*, 2014.

[105] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *CVPR*, 2015.

[106] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *CVPR*, 2016.

[107] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4700–4708.

[108] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint arXiv:1704.04861*, 2017.

[109] M. Tan and Q. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," in *International conference on machine learning*. PMLR, 2019, pp. 6105–6114.

[110] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein *et al.*, "Imagenet large scale visual recognition challenge," *International journal of computer vision*, 2015.

[111] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning*, 1995.

[112] L. Breiman, "Random forests," *Machine learning*, vol. 45, pp. 5–32, 2001.

[113] L. Van der Maaten and G. Hinton, "Visualizing data using t-sne." *Journal of machine learning research*, vol. 9, no. 11, 2008.

[114] M. Cimpoi, S. Maji, and A. Vedaldi, "Deep filter banks for texture recognition and segmentation," in *CVPR*, 2015.

[115] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 580–587.

[116] M. Mehri, P. Gomez-Krämer, P. Héroux, A. Boucher, and R. Mullot, "Texture feature evaluation for segmentation of historical document images," in *Proceedings of the 2nd International Workshop on Historical Document Imaging and Processing*, 2013, pp. 102–109.

[117] H. Tamura, S. Mori, and T. Yamawaki, "Textural features corresponding to visual perception," *IEEE Transactions on Systems, man, and cybernetics*, pp. 460–473, 1978.

[118] T. Ojala, M. Pietikainen, and D. Harwood, "Performance evaluation of texture measures with classification based on kullback discrimination of distributions," in *ICPR*, 1994, pp. 582–585.

[119] T. Ojala, M. Pietikäinen, and D. Harwood, "A comparative study of texture measures with classification based on featured distributions," *Pattern recognition*, pp. 51–59, 1996.

[120] X. Tang, "Texture information in run-length matrices," *IEEE transactions on image processing*, pp. 1602–1609, 1998.

[121] M. Petrou and P. G. Sevilla, *Image processing: dealing with texture*. Wiley Online Library, 2006, vol. 1.

[122] R. M. Haralick, K. Shanmugam *et al.*, "Textural features for image classification," *IEEE Transactions on systems, man, and cybernetics*, pp. 610–621, 1973.

[123] D. Gabor, "Theory of communication. part 1: The analysis of information," *Journal of the Institution of Electrical Engineers-Part III: Radio and Communication Engineering*, pp. 429–441, 1946.

[124] B. S. Manjunath and W.-Y. Ma, "Texture features for browsing and retrieval of image data," *IEEE Transactions on pattern analysis and machine intelligence*, pp. 837–842, 1996.

[125] A. Nicolaou, F. Slimane, V. Maergner, and M. Liwicki, "Local binary patterns for arabic optical font recognition," in *IAPR International Workshop on Document Analysis Systems, 2014*, 2014.

[126] V. Eglin, S. Bres, and C. Rivero, "Hermite and Gabor transforms for noise reduction and handwriting classification in ancient manuscripts," *International Journal of Document Analysis and Recognition (IJDAR)*, 2007.

[127] R. K. Hanusiak, L. S. Oliveira, E. Justino, and R. Sabourin, "Writer verification using texture-based features," *IJDAR*, 2012.

[128] S. Chaudhury and R. Sheth, "Trainable script identification strategies for indian languages," in *ICDAR*, 1999.

[129] M. A. Ferrer, A. Morales, and U. Pal, "LBP based line-wise script identification," in *International Conference on Document Analysis and Recognition (ICDAR), 2013*, 2013.

[130] T. Tan, "Rotation invariant texture features and their use in automatic script identification," *IEEE Transactions on pattern analysis and machine intelligence*, 1998.

[131] G. Peake and T. Tan, "Script and language identification from document images," in *WDIA*, 1997.

[132] J. Ding, L. Lam, and C. Y. Suen, "Classification of oriental and european scripts by using characteristic features," in *ICDAR*, 1997.

[133] D. Dhanya and A. Ramakrishnan, "Script identification in printed bilingual documents," in *IWDAS*, 2002.

[134] K. Chen, H. Wei, J. Hennebert, R. Ingold, and M. Liwicki, "Page segmentation for historical handwritten document images using color and texture features," in *ICFHR*, 2014.

[135] K. Chen, H. Wei, M. Liwicki, J. Hennebert, and R. Ingold, "Robust text line segmentation for historical manuscript images using color and texture," in *ICPR*, 2014, pp. 2978–2983.

[136] A. Garz and R. Sablatnig, "Multi-scale texture-based text recognition in ancient manuscripts," in *VSMM*, 2010, pp. 336–339.

[137] N. Nikolaou, M. Makridis, B. Gatos, N. Stamatopoulos, and N. Papamarkos, "Segmentation of historical machine-printed documents using adaptive run length smoothing and skeleton segmentation paths," *Image and Vision Computing*, pp. 590–604, 2010.

[138] A. Oyedotun, Oyebade K.and Khashman, "Document segmentation using textural features summarization and feedforward neural network," *Applied Intelligence*, 2016.

[139] M.-W. Lin, J.-R. Tapamo, and B. Ndovie, "A texture-based method for document segmentation and classification," *South African Computer Journal*, pp. 49–56, 2006.

[140] D. Coppi, C. Grana, and R. Cucchiara, "Illustrations segmentation in digitized documents using local correlation features," *Procedia Computer Science*, pp. 76–83, 2014.

[141] N. Chen and D. Blostein, "A survey of document image classification: problem statement, classifier architecture and performance evaluation," *IJDAR*, 2007.

[142] F. Alaei, A. Alaei, M. Blumenstein, and U. Pal, "Document image retrieval based on texture features and similarity fusion," in *International Conference on Image and Vision Computing New Zealand*, 2016, pp. 1–6.

[143] F. Alaei, A. Alaei, U. Pal, and M. Blumenstein, "Document image retrieval based on texture features: a recognition-free approach," in *2016 International Conference on Digital Image Computing: Techniques and Applications (DICTA)*. IEEE, 2016, pp. 1–7.

[144] A. Busch, W. W. Boles, and S. Sridharan, "Texture for script identification," *IEEE Trans.on PAMI*, 2005.

[145] W. Pan, C. Y. Suen, and T. D. Bui, "Script identification using steerable gabor filters," in *ICDAR*, 2005.

[146] V. Ablavsky and M. R. Stevens, "Automatic feature selection with applications to script identification of degraded documents." in *ICDAR*, 2003.

[147] P. Hiremath, J. D. Pujari, S. Shivashankar, and V. Mouneswara, "Script identification in a handwritten document image using texture features," in *IACC*, 2010.

[148] P. Hiremath and S. Shivashankar, "Wavelet based co-occurrence histogram features for texture classification with an application to script identification in a document image," *Pattern Recognition Letters*, 2008.

[149] M. Padma and P. Vijaya, "Entropy based texture features useful for automatic script identification," *International Journal on Computer Science and Engineering*, 2010.

[150] L. Zhou, X. J. Ping, E. Zheng, and L. Guo, "Script identification based on wavelet energy histogram moment features," in *ICSP*, 2010.

[151] H. Rezaee, M. Geravanchizadeh, and F. Razzazi, "Automatic language identification of bilingual english and farsi scripts," in *AICT*, 2009.

[152] S. Angadi and M. Kodabagi, "A fuzzy approach for word level script identification of text in low resolution display board images using wavelet features," in *ICACCI*, 2013.

[153] S. Chanda, O. R. Terrades, and U. Pal, "Svm based scheme for thai and english script identification," in *ICDAR*, 2007.

[154] S. Chanda and U. Pal, "English, devnagari and urdu text identification," in *Cognitive Recognition*, 2005.

[155] A. Busch, "Multi-font script identification using texture-based features," in *Image Analysis and Recognition*, A. Campilho and M. Kamel, Eds., 2006.

[156] P. Nagabhushan, S. Angadi, and B. Anami, "An intelligent pin code script identification methodology based on texture analysis using modified invariant moments," in *ICCR*, 2005.

[157] P. B. Pati, S. S. Raju, N. Pati, and A. Ramakrishnan, "Gabor filters for document analysis in indian bilingual documents," in *ICISIP*, 2004.

[158] V. Singhal, N. Navin, and D. Ghosh, "Script-based classification of hand-written text documents in a multilingual environment," in *RIDE-MLIM*, 2003.

[159] S. Jaeger, H. Ma, and D. Doermann, "Identifying script on word-level with informational confidence," in *ICDAR*, 2005.

[160] P. B. Pati and A. Ramakrishnan, "Word level multi-script identification," *Pattern Recognition Letters*, 2008.

[161] R. S. Kunte and R. D. S. Samuel, "On separation of Kannada and English words from a bilingual document employing Gabor features and radial basis function neural network," *ICCR*, 2005.

[162] B. Philip and R. S. Samuel, "A novel bilingual OCR for printed Malayalam-English text based on Gabor features and dominant singular values," in *ICDIP*, 2009.

[163] R. Rani, R. Dhir, and G. S. Lehal, "Script identification of pre-segmented multi-font characters and digits," in *ICDAR*, 2013.

[164] S. Chanda, K. Franke, and U. Pal, "Identification of indic scripts on torn-documents," in *ICDAR*, 2011.

[165] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *NIPS*, 2012, pp. 1097–1105.

[166] A. W. Harley, A. Ufkes, and K. G. Derpanis, "Evaluation of deep convolutional nets for document image classification and retrieval," in *ICDAR*, 2015.

[167] C. Tensmeyer and T. Martinez, "Analysis of convolutional neural networks for document image classification," in *ICDAR*, 2017.

[168] A. Kölsch, M. Z. Afzal, M. Ebbecke, and M. Liwicki, "Real-time document image classification using deep CNN and extreme learning machines," *CoRR*, 2017.

[169] K. Chen and M. Seuret, "Convolutional neural networks for page segmentation of historical document images," *CoRR*, 2017.

[170] C. Wick and F. Puppe, "Fully convolutional neural networks for page segmentation of historical document images," in *IWDAS*, 2018.

[171] M. Alberti, M. Seuret, V. Pondenkandath, R. Ingold, and M. Liwicki, "Historical document image segmentation with lda-initialized deep neural networks," *CoRR*, 2017.

[172] Q. N. Vo, S. H. Kim, H. J. Yang, and G. Lee, "Binarization of degraded document images based on hierarchical deep supervised network," *Pattern Recognition*, 2018.

[173] K. Zagoris and I. Pratikakis, "Bio-inspired modeling for the enhancement of historical handwritten documents," in *ICDAR*, 2017.

[174] X. Peng, H. Cao, and P. Natarajan, "Using convolutional encoder-decoder for document image binarization," in *ICDAR*, 2017.

[175] C. Tensmeyer and T. Martinez, "Document image binarization with fully convolutional neural networks," *CoRR*, 2017.

[176] J. Calvo-Zaragoza and A.-J. Gallego, "A selectional auto-encoder approach for document image binarization," *CoRR*, 2017.

[177] O. G. Cula and K. J. Dana, "Compact representation of bidirectional texture functions," in *CVPR*, 2001.

[178] K. Chen and M. Seuret, "Convolutional neural networks for page segmentation of historical document images," in *ICDAR*, 2017.

[179] T. Leung and J. Malik, "Representing and recognizing the visual appearance of materials using three-dimensional textons," *IJCV*, 2001.

[180] A. Bruno, L. Greco, and M. La Cascia, "Video object recognition and modeling by sift matching optimization." in *ICPRAM*, 2014.

[181] J. Yang, Y.-G. Jiang, A. G. Hauptmann, and C.-W. Ngo, "Evaluating bag-of-visual-words representations in scene classification," in *Workshop on multimedia information retrieval*, 2007.

[182] G.-H. Liu, L. Zhang, Y.-K. Hou, Z.-Y. Li, and J.-Y. Yang, "Image retrieval based on multi-texton histogram," *Pattern Recognition*, 2010.

[183] M. Cimpoi, S. Maji, I. Kokkinos, S. Mohamed, and A. Vedaldi, "Describing textures in the wild," in *CVPR*, 2014.

[184] H. Zhang, J. Xue, and K. Dana, "Deep ten: Texture encoding network," in *CVPR*, 2017.

[185] H. Zhang, T. Xu, M. Elhoseiny, X. Huang, S. Zhang, A. Elgammal, and D. Metaxas, "Spda-cnn: Unifying semantic part detection and abstraction for fine-grained recognition," in *CVPR*, 2016.

[186] S. Huang, Z. Xu, D. Tao, and Y. Zhang, "Part-stacked cnn for fine-grained visual categorization," in *CVPR*, 2016.

[187] J. Fu, H. Zheng, and T. Mei, "Look closer to see better: Recurrent attention convolutional neural network for fine-grained image recognition," in *CVPR*, 2017.

[188] H. Zheng, J. Fu, T. Mei, and J. Luo, "Learning multi-attention convolutional neural network for fine-grained image recognition," in *CVPR*, 2017.

[189] Y. Wang, V. I. Morariu, and L. S. Davis, "Learning a discriminative filter bank within a cnn for fine-grained recognition," in *CVPR*, 2018.

[190] E. Appiani, F. Cesarini, A. M. Colla, M. Diligenti, M. Gori, S. Marinai, and G. Soda, "Automatic document classification and indexing in high-volume applications," *IJDAR*, 2001.

[191] A. D. Bagdanov and M. Worring, "Fine-grained document genre classification using first order random graphs," in *DAR*, 2001.

[192] L. Kang, J. Kumar, P. Ye, Y. Li, and D. Doermann, "Convolutional neural networks for document image classification," in *ICPR*, 2014.

[193] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman, "Return of the devil in the details: Delving deep into convolutional nets," *arXiv preprint arXiv:1405.3531*, 2014.

[194] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *CVPR*, 2016.

[195] C. A. Mello and R. D. Lins, "Image segmentation of historical documents," *Visual2000, Mexico City, Mexico*, vol. 30, 2000.

[196] Z. Shi and V. Govindaraju, "Historical document image segmentation using background light intensity normalization," in *Document Recognition and Retrieval XII*, vol. 5676. SPIE, 2005, pp. 167–174.

[197] K. Chen, M. Seuret, M. Liwicki, J. Hennebert, and R. Ingold, "Page segmentation of historical document images with convolutional autoencoders," in *ICDAR*, 2015.

[198] K. Chen, C.-L. Liu, M. Seuret, M. Liwicki, J. Hennebert, and R. Ingold, "Page segmentation for historical document images based on superpixel classification with unsupervised feature learning," in *2016 12th IAPR Workshop on Document Analysis Systems (DAS)*. IEEE, 2016, pp. 299–304.

[199] K. Chen, M. Seuret, M. Liwicki, J. Hennebert, C. Liu, and R. Ingold, "Page segmentation for historical handwritten document images using conditional random fields," in *ICFHR*, 2016.

[200] J. Lafferty, A. McCallum, and F. C. Pereira, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," *ICML*, 2001.

[201] A. Amin and R. Shiu, "Page segmentation and classification utilizing bottom-up approach," *IJIG*, 2001.

[202] J. Ha, R. M. Haralick, and I. T. Phillips, "Recursive xy cut using bounding boxes of connected components," in *ICDAR*, 1995.

[203] Q. N. Vo and G. Lee, "Dense prediction for text line segmentation in handwritten document images," in *ICIP*, 2016.

[204] G. Renton, C. Chatelain, S. Adam, C. Kermorvant, and T. Paquet, "Handwritten text line segmentation using fully convolutional network," in *ICDAR*, 2017.

[205] X. Yang, E. Yumer, P. Asente, M. Kraley, D. Kifer, and C. L. Giles, "Learning to extract semantic structure from documents using multimodal fully convolutional neural networks," in *CVPR*, 2017.

[206] A. Vil'kin, I. Safonov, and M. Egorova, "Bottom-up document segmentation method based on textural features," *PRIA*, 2011.

[207] G. Louloudis, B. Gatos, I. Pratikakis, and C. Halatsis, "Text line and word segmentation of handwritten documents," *Pattern Recognition*, pp. 3169–3183, 2009.

[208] V. Papavassiliou, T. Stafylakis, V. Katsouros, and G. Carayannis, "Handwritten document image segmentation into text lines and words," *Pattern Recognition*, pp. 369–377, 2010.

[209] S. Mao, A. Rosenfeld, and T. Kanungo, "Document structure analysis algorithms: a literature survey," in *DRR*, 2003.

[210] R. M. Haralick, "Document image understanding: Geometric and logical layout," in *CVPR*, 1994.

[211] M. Shilman, P. Liang, and P. Viola, "Learning nongenerative grammatical models for document analysis," in *ICCV*, 2005.

[212] M. Mehri, P. Héroux, P. Gomez-Krämer, and R. Mullot, "Texture feature benchmarking and evaluation for historical document image analysis," *IJDAR*, 2017.

[213] T. Lu and A. Dooms, "Probabilistic homogeneity for document image segmentation," *Pattern Recognition*, pp. 107 591–107 605, 2021.

[214] X.-H. Li, F. Yin, and C.-L. Liu, "Page segmentation using convolutional neural network and graphical model," in *DAS*, 2020, pp. 231–245.

[215] G. Csurka, D. Larlus, A. Gordo, and J. Almazan, "What is the right way to represent document images?" *arXiv*, 2016.

[216] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, "Inception-v4, inception-resnet and the impact of residual connections on learning," in *Thirty-first AAAI conference on artificial intelligence*, 2017.

[217] R. Sarkhel and A. Nandi, "Deterministic routing between layout abstractions for multi-scale classification of visually rich documents," in *IJCAI*, 2019.

[218] Y. Xu, M. Li, L. Cui, S. Huang, F. Wei, and M. Zhou, "Layoutlm: Pre-training of text and layout for document image understanding," in *ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2020.

[219] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv*, 2018.

[220] T. Dauphinee, N. Patel, and M. Rashidi, "Modular multimodal architecture for document classification," *arXiv*, 2019.

[221] S. Appalaraju, B. Jasani, B. U. Kota, Y. Xie, and R. Manmatha, "Docformer: End-to-end transformer for document understanding," in *ICCV*, 2021.

[222] J. Zujovic, L. Gandy, S. Friedman, B. Pardo, and T. N. Pappas, "Classifying paintings by artistic genre: An analysis of features & classifiers," in *International Workshop on Multimedia Signal Processing*, 2009.

[223] H. Chiang, Y. Ge, and C. Wu, "Classification of book genres by cover and title," 2015.

[224] G. R. Biradar, J. Raagini, A. Varier, and M. Sudhir, "Classification of book genres using book cover and title," in *International Conference on Intelligent Systems and Green Technology (ICISGT)*, 2019.

[225] A. Lucieri, H. Sabir, S. A. Siddiqui, S. T. R. Rizvi, B. K. Iwana, S. Uchida, A. Dengel, and S. Ahmed, "Benchmarking deep learning models for classification of book covers," *SN Computer Science*, 2020.

[226] Y. Liu, X. Lu, Y. Qin, Z. Tang, and J. Xu, "Review of chart recognition in document images," in *VDA*, 2013.

[227] I. Kavasidis, S. Palazzo, C. Spampinato, C. Pino, D. Giordano, D. Giuffrida, and P. Messina, "A saliency-based convolutional neural network for table and chart detection in digitized documents," *arXiv preprint arXiv:1804.06236*, 2018.

[228] N. Aletras and A. Mittal, "Labeling topics with images using a neural network," in *European Conference on Information Retrieval*, 2017.

[229] J. Charbonnier, L. Sohmen, J. Rothman, B. Rohden, and C. Wartena, "Noa: A search engine for reusable scientific images beyond the life sciences," in *European Conference on Information Retrieval*, 2018.

[230] L. Shijian and C. L. Tan, "Script and language identification in noisy and degraded document images," *IEEE Trans. on PAMI*, 2007.

[231] L. Zhou, Y. Lu, and C. L. Tan, "Bangla/english script identification based on analysis of connected component profiles," in *International Workshop on Document Analysis Systems*, 2006.

[232] N. Sharma, U. Pal, and M. Blumenstein, "A study on word-level multi-script identification from video frames," in *2014 International Joint Conference on Neural Networks (IJCNN)*, 2014.

[233] J. Mei, L. Dai, B. Shi, and X. Bai, "Scene text script identification with convolutional recurrent neural networks," in *ICPR*, 2016.

[234] L. Lu, Y. Yi, F. Huang, K. Wang, and Q. Wang, "Integrating local cnn and global cnn for script identification in natural scene images," *IEEE Access*, 2019.

[235] A. K. Bhunia, A. Konwer, A. K. Bhunia, A. Bhowmick, P. P. Roy, and U. Pal, "Script identification in natural scene image and video frames using an attention based convolutional-LSTM network," *Pattern Recognition*, 2019.

[236] M. Ghosh, H. Mukherjee, S. M. Obaidullah, K. Santosh, N. Das, and K. Roy, "Lwsinet: A deep learning-based approach towards video script identification," *Multimedia Tools and Applications*, 2021.

[237] H. Noh, S. Hong, and B. Han, "Learning deconvolution network for semantic segmentation," in *CVPR*, 2015.

[238] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid scene parsing network," in *CVPR*, 2017.

[239] A. Garcia-Garcia, S. Orts-Escolano, S. Oprea, V. Villena-Martinez, P. Martinez-Gonzalez, and J. Garcia-Rodriguez, "A survey on deep learning techniques for image and video semantic segmentation," *Applied Soft Computing*, pp. 41–65, 2018.

[240] B. Li, Y. Shi, Z. Qi, and Z. Chen, "A survey on semantic segmentation," in *ICDMW*, 2018, pp. 1233–1240.

[241] S. Hao, Y. Zhou, and Y. Guo, "A brief survey on semantic segmentation with deep learning," *Neurocomputing*, pp. 302–321, 2020.

[242] Z. Shen, R. Zhang, M. Dell, B. C. G. Lee, J. Carlson, and W. Li, "Layoutparser: A unified toolkit for deep learning based document image analysis," in *Document Analysis and Recognition–ICDAR 2021: 16th International Conference, Lausanne, Switzerland, September 5–10, 2021, Proceedings, Part I 16*. Springer, 2021, pp. 131–146.

[243] Z. Shen, K. Zhang, and M. Dell, "A large dataset of historical japanese documents with complex layouts," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2020, pp. 548–549.

[244] B. C. G. Lee, J. Mears, E. Jakeway, M. Ferriter, C. Adams, N. Yarasavage, D. Thomas, K. Zwaard, and D. S. Weld, "The newspaper navigator dataset: extracting and analyzing visual content from 16 million historic newspaper pages in chronicling america," *arXiv preprint arXiv:2005.01583*, 2020.

[245] M. Li, L. Cui, S. Huang, F. Wei, M. Zhou, and Z. Li, "Tablebank: Table benchmark for image-based table detection and recognition," in *Proceedings of the Twelfth Language Resources and Evaluation Conference*, 2020, pp. 1918–1925.

[246] T. Shehzadi, D. Stricker, and M. Z. Afzal, "A hybrid approach for document layout analysis in document images," in *International Conference on Document Analysis and Recognition*. Springer, 2024, pp. 21–39.

[247] J. Wang, K. Hu, and Q. Huo, "Dlaformer: An end-to-end transformer for document layout analysis," in *International Conference on Document Analysis and Recognition*. Springer, 2024, pp. 40–57.

[248] Y. Xu, Y. Xu, T. Lv, L. Cui, F. Wei, G. Wang, Y. Lu, D. Florencio, C. Zhang, W. Che *et al.*, "Layoutlmv2: Multi-modal pre-training for visually-rich document understanding," *arXiv*, 2020.

[249] Y. Huang, T. Lv, L. Cui, Y. Lu, and F. Wei, "Layoutlmv3: Pre-training for document ai with unified text and image masking," in *Proceedings of the 30th ACM International Conference on Multimedia*, 2022, pp. 4083–4091.

[250] Z. Tang, Z. Yang, G. Wang, Y. Fang, Y. Liu, C. Zhu, M. Zeng, C. Zhang, and M. Bansal, "Unifying vision, text, and layout for universal document processing," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2023, pp. 19 254–19 264.

[251] Z. Gu, C. Meng, K. Wang, J. Lan, W. Wang, M. Gu, and L. Zhang, "Xylayoutlm: Towards layout-aware multimodal networks for visually-rich document understanding," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 4583–4592.

[252] S. Maity, S. Biswas, S. Manna, A. Banerjee, J. Lladós, S. Bhattacharya, and U. Pal, "Selfdocseg: A self-supervised vision-based approach towards document segmentation," in *International Conference on Document Analysis and Recognition*. Springer, 2023, pp. 342–360.

[253] M. Auvray, S. Hanneton, and J. K. O'Regan, "Learning to perceive with a visuo—auditory substitution system: localisation and object recognition with 'the voice'," *Perception*, 2007.

[254] S. Meers and K. Ward, "A vision system for providing 3d perception of the environment via transcutaneous electro-neural stimulation," in *ICIV*, 2004.

[255] J. Bai, Z. Liu, Y. Lin, Y. Li, S. Lian, and D. Liu, "Wearable travel aid for environment perception and navigation of visually impaired people," *Electronics*, 2019.

[256] B. Makav and V. Kılıç, "Smartphone-based image captioning for visually and hearing impaired," in *ELECO*, 2019.

[257] S. Singh, S. Choudhury, K. Vishal, and C. Jawahar, "Currency recognition on mobile phones," in *ICPR*. IEEE, 2014.

[258] S. Wu, J. Wieland, O. Farivar, and J. Schiller, "Automatic alt-text: Computer-generated image descriptions for blind users on a social network service," in *ACM Conf. on CSCWSC*, 2017.

[259] R. Shilkrot, J. Huber, W. Meng Ee, P. Maes, and S. C. Nanayakkara, "Fingerreader: a wearable device to explore printed text on the go," in *ACM Conf. on HFCS*, 2015.

[260] G. Larsson, M. Maire, and G. Shakhnarovich, "Learning representations for automatic colorization," in *ECCV*, 2016.

[261] R. Zhang, P. Isola, and A. A. Efros, "Colorful image colorization," in *ECCV*, 2016.

[262] P. Isola, J. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *CVPR*, 2017.

[263] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *ICCV*, 2017.

[264] Z. Yi, H. Zhang, P. Tan, and M. Gong, "Dualgan: Unsupervised dual learning for image-to-image translation," in *ICCV*, 2017.

[265] S. G. Deshpande and J.-N. Hwang, "A real-time interactive virtual classroom multimedia distance learning system," *IEEE Transactions on multimedia*, 2001.

[266] Z. Zhu, C. McKittrick, and W. Li, "Virtualized classroom-automated production, media integration and user-customized presentation," in *2004 Conference on Computer Vision and Pattern Recognition Workshop*, 2004.

[267] S. Bulathwela, M. Perez-Ortiz, E. Yilmaz, and J. Shawe-Taylor, "Vlengagement: A dataset of scientific video lectures for evaluating population-based engagement," *arXiv preprint arXiv:2011.02273*, 2020.

[268] I. Li, A. R. Fabbri, R. R. Tung, and D. R. Radev, "What should i learn first: Introducing lecturebank for nlp education and prerequisite chain learning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, 2019, pp. 6674–6681.

[269] D. Galanopoulos and V. Mezaris, "Temporal lecture video fragmentation using word embeddings," in *MultiMedia Modeling: 25th International Conference, MMM 2019, Thessaloniki, Greece, January 8–11, 2019, Proceedings, Part II 25*. Springer, 2019, pp. 254–265.

[270] K. Dutta, M. Mathew, P. Krishnan, and C. Jawahar, "Localizing and recognizing text in lecture videos," in *2018 16th international conference on frontiers in handwriting recognition (ICFHR)*. IEEE, 2018, pp. 235–240.

[271] H. Chen, M. Cooper, D. Joshi, and B. Girod, "Multi-modal language models for lecture video retrieval," in *Proceedings of the 22nd ACM international conference on Multimedia*, 2014, pp. 1081–1084.

[272] N. Van Nguyen, M. Coustaty, and J.-M. Ogier, "Multi-modal and cross-modal for lecture videos retrieval," in *2014 22nd International Conference on Pattern Recognition*. IEEE, 2014, pp. 2667–2672.

[273] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark *et al.*, "Learning transferable visual models from natural language supervision," in *International conference on machine learning*, 2021.

[274] S. Chun, S. J. Oh, R. S. De Rezende, Y. Kalantidis, and D. Larlus, "Probabilistic embeddings for cross-modal retrieval," in *CVPR*, 2021.

[275] Y. Song and M. Soleymani, "Polysemous visual-semantic embedding for cross-modal retrieval," in *CVPR*, 2019.

[276] S. J. Oh, K. Murphy, J. Pan, J. Roth, F. Schroff, and A. Gallagher, "Modeling uncertainty with hedged instance embedding," *arXiv preprint arXiv:1810.00319*, 2018.

[277] A. A. Alemi, I. Fischer, J. V. Dillon, and K. Murphy, "Deep variational information bottleneck," *arXiv preprint arXiv:1612.00410*, 2016.

[278] L. Liu, Z. Wang, T. Qiu, Q. Chen, Y. Lu, and C. Y. Suen, "Document image classification: Progress over two decades," *Neurocomputing*, 2021.

[279] H. Zheng, J. Fu, Z.-J. Zha, and J. Luo, "Looking for the devil in the details: Learning trilinear attention sampling network for fine-grained image recognition," in *CVPR*, 2019.

[280] I. Loshchilov and F. Hutter, "Sgdr: Stochastic gradient descent with warm restarts," *arXiv*, 2016.

[281] P. Li, J. Gu, J. Kuen, V. I. Morariu, H. Zhao, R. Jain, V. Manjunatha, and H. Liu, "Selfdoc: Self-supervised document representation learning," in *CVPR*, 2021.

[282] H. Bao, L. Dong, F. Wei, W. Wang, N. Yang, X. Liu, Y. Wang, J. Gao, S. Piao, M. Zhou *et al.*, "Unilmv2: Pseudo-masked language models for unified language model pre-training," in *International Conference on Machine Learning*. PMLR, 2020.

[283] S. Ukil, S. Ghosh, S. Md Obaidullah, K. Santosh, K. Roy, and N. Das, "Deep learning for word-level handwritten indic script identification," *CoRR*, 2018.

[284] N. Sharma, R. Mandal, R. Sharma, U. Pal, and M. Blumenstein, "ICDAR2015 competition on video script identification (CVSI 2015)," in *ICDAR*, 2015.

[285] Y. Zhong, K. Karu, and A. K. Jain, "Locating text in complex color images," *Pattern recognition*, vol. 28, no. 10, pp. 1523–1535, 1995.

[286] F. Simistira, M. Seuret, N. Eichenberger, A. Garz, M. Liwicki, and R. Ingold, "Diva-hisdb: A precisely annotated large dataset of challenging medieval manuscripts," in *2016 15th International Conference on Frontiers in Handwriting Recognition (ICFHR)*. IEEE, 2016, pp. 471–476.

[287] Y. Ganin and V. Lempitsky, "-fields: neural network nearest neighbor fields for image transforms," in *Asian conference on computer vision*. Springer, 2014, pp. 536–551.

[288] K. Chen, M. Seuret, M. Liwicki, J. Hennebert, C.-L. Liu, and R. Ingold, "Page segmentation for historical handwritten document images using conditional random fields," in *2016 15th International Conference on Frontiers in Handwriting Recognition (ICFHR)*. IEEE, 2016, pp. 90–95.

[289] K. Chen, M. Seuret, J. Hennebert, and R. Ingold, "Convolutional neural networks for page segmentation of historical document images," in *ICDAR*, 2017.

[290] A. Fischer, A. Keller, V. Frinken, and H. Bunke, "Lexicon-free handwritten word spotting using character hmms," *Pattern recognition letters*, vol. 33, no. 7, pp. 934–942, 2012.

[291] A. Fischer, V. Frinken, A. Fornés, and H. Bunke, "Transcription alignment of latin manuscripts using hidden markov models," in *Proceedings of the 2011 Workshop on Historical Document Imaging and Processing*, 2011, pp. 29–36.

[292] A. Fischer, M. Wuthrich, M. Liwicki, V. Frinken, H. Bunke, G. Viehhauser, and M. Stolz, "Automatic transcription of handwritten medieval documents," in *2009 15th International Conference on Virtual Systems and Multimedia*. IEEE, 2009, pp. 137–142.

[293] T. Leung and J. Malik, "Recognizing surfaces using three-dimensional textons," in *ICCV*, 1999.

[294] B. Julesz and J. R. Bergen, "Human factors and behavioral science: Textons, the fundamental elements in preattentive vision and perception of textures," *Bell system technical journal*, vol. 62, no. 6, pp. 1619–1645, 1983.

[295] L. Liu, L. Wang, and X. Liu, "In defense of soft-assignment coding," in *2011 International Conference on Computer Vision*, 2011.

[296] F. Perronnin, J. Sánchez, and T. Mensink, "Improving the fisher kernel for large-scale image classification," in *ECCV*, 2010.

[297] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *ECCV*, 2014.

[298] S. A. Oliveira, B. Seguin, and F. Kaplan, "dhsegment: A generic deep-learning approach for document segmentation," in *2018 16th International Conference on Frontiers in Handwriting Recognition (ICFHR)*. IEEE, 2018, pp. 7–12.

[299] Y. Xu, F. Yin, Z. Zhang, C.-L. Liu *et al.*, "Multi-task layout analysis for historical handwritten documents using fully convolutional networks." in *IJCAI*, 2018, pp. 1057–1063.

[300] L. Studer, M. Alberti, V. Pondenkandath, P. Goktepe, T. Kolonko, A. Fischer, M. Liwicki, and R. Ingold, "A comprehensive study of imagenet pre-training for historical document image analysis," in *2019 International Conference on Document Analysis and Recognition (ICDAR)*. IEEE, 2019, pp. 720–725.

[301] L. v. d. Maaten and G. Hinton, "Visualizing data using t-sne," *Journal of machine learning research*, 2008.

[302] W. Liu, A. Rabinovich, and A. C. Berg, "Parsenet: Looking wider to see better," *arXiv preprint arXiv:1506.04579*, 2015.

[303] R. Liu, J. Lehman, P. Molino, F. P. Such, E. Frank, A. Sergeev, and J. Yosinski, "An intriguing failing of convolutional neural networks and the coordconv solution," *arXiv preprint arXiv:1807.03247*, 2018.

[304] F. Yu and V. Koltun, "Multi-scale context aggregation by dilated convolutions," *arXiv*, 2015.

[305] L. Todoran, M. Worring, and A. W. Smeulders, "The UvA color document dataset," *IJDAR*, 2005.

[306] D. Rolnick, A. Veit, S. Belongie, and N. Shavit, "Deep learning is robust to massive label noise," *arXiv*, 2017.

[307] C. Clark and S. Divvala, "Pdffigures 2.0: Mining figures from research papers," in *Digital Libraries (JCDL)*, 2016.

[308] J. C. Schlimmer and D. Fisher, "A case study of incremental concept induction," in *AAAI*, 1986.

[309] A. Dutta and A. Zisserman, "The VIA annotation software for images, audio and video," in *Proceedings of the 27th ACM International Conference on Multimedia*, ser. MM '19. New York, NY, USA: ACM, 2019. [Online]. Available: https://doi.org/10.1145/3343031.3350535

[310] M. Parmar, S. Mishra, M. Geva, and C. Baral, "Don't blame the annotator: Bias already starts in the annotation instructions," *arXiv preprint arXiv:2205.00415*, 2022.

[311] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell *et al.*, "Language models are few-shot learners," *Advances in neural information processing systems*, vol. 33, pp. 1877–1901, 2020.

[312] M. Li, Z. Lin, R. Mech, E. Yumer, and D. Ramanan, "Photo-sketching: Inferring contour drawings from images," in *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE, 2019, pp. 1403–1412.

[313] K. V. Jobin, A. Mondal, and C. V. Jawahar, "Document image analysis using deep multi-modular features," *SN Computer Science*, vol. 4, no. 1, p. 5, 2022.

[314] M. Grootendorst, "Keybert: Minimal keyword extraction with bert," *Zenodo*, 2020.

[315] S. Lohmann, F. Heimerl, F. Bopp, M. Burch, and T. Ertl, "Concentri cloud: Word cloud visualization for multiple text documents," in *In International Conference on Information Visualisation*, 2015.

[316] W. Lin, F. Zhang, R. Lian, L. Xu, X. Chen, and L. Kuang, "Seam carving algorithm based on saliency," in *ICSVTTCA*, 2017.

[317] T. Yin, G. Yang, L. Li, D. Zhang, and X. Sun, "Detecting seam carving based image resizing using local binary patterns," *Computers & Security*, 2015.

[318] M.-T. Luong, H. Pham, and C. D. Manning, "Effective approaches to attention-based neural machine translation," *arXiv*, 2015.

[319] X. Yi, L. Gao, Y. Liao, X. Zhang, R. Liu, and Z. Jiang, "CNN based page object detection in document images," in *ICDAR*, 2017.

[320] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, 1997.

[321] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," in *ICLR*, 2015.

[322] R. Smith, "An overview of the tesseract ocr engine," in *ICDAR*, 2007.

[323] A. Sachin Raja, Mondal and C. V. Jawahar, "Table structure recognition using top-down and bottom-up cues," in *ECCV*, 2020.

[324] E. Murphy, R. Kuber, G. McAllister, P. Strain, and W. Yu, "An empirical investigation into the difficulties experienced by visually impaired internet users," *UAIS*, 2008.

[325] W. Kim, B. Son, and I. Kim, "Vilt: Vision-and-language transformer without convolution or region supervision," in *International Conference on Machine Learning*. PMLR, 2021, pp. 5583–5594.

[326] K. V. Jobin, A. Mishra, and C. V. Jawahar, "Semantic labels-aware transformer model for searching over a large collection of lecture-slides," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2024, pp. 6016–6025.

[327] T. G. Dietterich, R. H. Lathrop, and T. Lozano-Pérez, "Solving the multiple instance problem with axis-parallel rectangles," *Artificial intelligence*, vol. 89, no. 1-2, pp. 31–71, 1997.

[328] A. Gretton, K. Borgwardt, M. Rasch, B. Schölkopf, and A. Smola, "A kernel method for the two-sample-problem," *Advances in neural information processing systems*, vol. 19, 2006.