applied sciences

MDPI

Article

Content-Based Video Big Data Retrieval with Extensive Features and Deep Learning

Thuong-Cang Phan [1,*], Anh-Cang Phan [2,*], Hung-Phi Cao [2] and Thanh-Ngoan Trieu [1,3]

[1] College of Information and Communication Technology, Can Tho University, Can Tho 94115, Vietnam; ngoan.trieuthanh@etudiant.univ-brest.fr
[2] Faculty of Information Technology, Vinh Long University of Technology Education, Vinh Long 85110, Vietnam; caohungphi@vlute.edu.vn
[3] La Faculté Sciences et Techniques, Université de Bretagne Occidentale, 29200 Brest, France
* Correspondence: ptcang@cit.ctu.edu.vn (T.-C.P.); cangpa@vlute.edu.vn (A.-C.P.); Tel.: +84-292-3831-301 (T.-C.P.); +84-918-204-917 (A.-C.P.)

Abstract: In the era of digital media, the rapidly increasing volume and complexity of multimedia data cause many problems in storing, processing, and querying information in a reasonable time. Feature extraction and processing time play an extremely important role in large-scale video retrieval systems and currently receive much attention from researchers. We, therefore, propose an efficient approach to feature extraction on big video datasets using deep learning techniques. It focuses on the main features, including subtitles, speeches, and objects in video frames, by using a combination of three techniques: optical character recognition (OCR), automatic speech recognition (ASR), and object identification with deep learning techniques. We provide three network models developed from networks of Faster R-CNN ResNet, Faster R-CNN Inception ResNet V2, and Single Shot Detector MobileNet V2. The approach is implemented in Spark, the next-generation parallel and distributed computing environment, which reduces the time and space costs of the feature extraction process. Experimental results show that our proposal achieves an accuracy of 96% and a processing time reduction of 50%. This demonstrates the feasibility of the approach for content-based video retrieval systems in a big data context.

Keywords: video retrieval; deep learning; spark; big data; video content extraction; content-based video big data

Citation: Phan, T.-C.; Phan, A.-C.; Cao, H.-P.; Trieu, T.-N. Content-Based Video Big Data Retrieval with Extensive Features and Deep Learning. *Appl. Sci.* **2022**, *12*, 6753. https://doi.org/10.3390/app12136753

Academic Editors: Yang-Lang Chang, Mohammad Alkhaleefah and Tan-Hsu Tan

Received: 13 June 2022
Accepted: 1 July 2022
Published: 3 July 2022

1. Introduction

With the development of the Internet, big data, and broadband networks, the demand for multimedia data in visualization is growing rapidly, and thus, multimedia information systems are increasingly important. However, multimedia data requires large amounts of storage and processing. Therefore, there is a need to efficiently extract, index, store, and retrieve video information from a large multimedia database. Video is one of the most common information transmissions and is easily accessible to many users around the world because of its visual and vivid advantages. A challenge posed to multimedia managers is to leverage the video content for storing and querying videos. The current common search engines often rely on titles and basic information about the videos and ignore content-intensive searches. Internet users demand to be able to query accurate videos in near real time as an alternative to traditional methods of keyword search. Videos are electronic continuous mediums; thus, storing and converting videos gives a much bigger challenge than normal text data [1]. In recent years, many methods have been developed for feature extraction to index and retrieve videos based on their content features. The video content features can be extracted consisting of the object, motion, speech, etc. Abderrahmane Adoui et al. proposed to use spatio-temporal features such as motion direction to characterize and compare video sub-sequences [2]. Simone Accattoli et al. used

a 3D convolutional neural network (CNN) to detect aggressive motions and violent scenes in video streams [3]. Donghuo Zeng et al. presented their approach for cross-modal video retrieval using supervised deep canonical correlation analysis to project audio and video into a shared space bridging their semantic gaps [4].

In this paper, we provide a feature extraction method for indexing and retrieving content-based videos in the distributed and parallel computing environment of Spark. The content features extracted from videos consist of subtitles, speeches, and objects existing in video frames, which are usually used in video queries. In order to achieve this, we use the techniques for recognizing optical characters, speeches, and objects using distributed deep learning in Spark. This approach builds machine learning models that run across multiple computing nodes to take advantage of distributed storage and computation. Spark's in-memory processing capabilities significantly reduce the cost of data transmission across the network and increase processing speed. Optical character recognition (OCR) is used to extract features from subtitles, while automatic speech recognition (ASR) is used for speech feature extraction in videos. Besides, deep neural networks have achieved impressive performance in many fields, such as image classification, object detection, and semantic segmentation. Jonathan Huang et al. provided a brief survey of modern convolution systems and compared the accuracy/speed of the architectures, including Faster R-CNN, Single Shot Detector, and R-FCN [5]. The research has shown that Faster R-CNN using Inception ResNet delivers the highest accuracy at one frame per second (FPS) for all tested cases; SSD on MobileNet has the highest mAP (mean average precision—Section 3.5.2) among the network models. Thus, distributed deep learning is used in this study to extract features from objects in video frames with three proposed network models developed from networks of Faster R-CNN ResNet, Faster R-CNN Inception ResNet V2, and Single Shot Detector MobileNet V2. In addition, we used the transfer learning technique to shorten the training time. The extracted features are represented in text format. They are then indexed and stored on a Hadoop Distributed File System (HDFS) [6] for querying purposes. The feature extraction is implemented in the Spark distributed environment to decrease the time and space costs. Experimental results of several scenarios show that the proposed method has an accuracy of 96% and the processing time is shortened by over 50%. We also provide a comparison of normal video extraction and distributed video extraction to demonstrate the efficiency of our approach. The approach can extract suitable video features for big data-driven video retrieval systems. It improves the system performance in terms of computing time and solves the problem of computing resource limitations. We take advantage of deep learning techniques and improve the network models to have appropriate models for video features extraction. The proposed approach contributes to optimize data processing in big multimedia management systems. Moreover, the novel contribution in this work is adapting to the trend of the modern knowledge management system, ensuring a complete knowledge development process, including knowledge exploration and exploitation. The multimedia knowledge management system will help the process of converting information into knowledge in a systematic way, as suggested in the research [7].

The structure of this paper is organized as follows. Section 2 presents the related works of the current problem and the general information of the techniques used in this work introduced in Section 3. Section 4 provides our proposed approach to content-based video retrieval systems. Section 5 presents the experiments with several scenarios and the comparison between the scenarios is provided in Section 6. The conclusion of the paper is presented in Section 7.

2. State of the Art Overview

In recent years, many studies on video retrieval have been conducted. These studies have great improvements in accuracy and processing speed but are still limited in the diversity of content recognition and experimental datasets. Yu Youngjae et al. proposed a word detector from video input that did not require external knowledge sources for training

and was trainable in an end-to-end manner jointly with any video-to-language models [8]. The proposal was demonstrated in the Large Scale Movie Description Challenge 2016. Pavlos Avgoustinakis et al. addressed the problem of audio-based near-duplicate video retrieval by capturing temporal patterns of audio similarity between video pairs [9]. They used a pre-trained CNN on a large scale dataset of audio events and calculated the similarity matrix derived from the pairwise similarity of these descriptors. The experiments were conducted on three visual-based datasets, i.e., FIVR-200K [10], SVD [11], and EVVE [12]. In 2019, Pandeya and Lee applied deep learning to classify emotions based on music videos [13]. This system was tested on four unimodal and four multimodal neural networks and the best model had an accuracy of 88.56%. This method automatically classified human high-level emotions with relatively high performance. This study focused on processing musical video features in a traditional environment and did not consider distributed and parallel processing in a big data context. M. Braveen proposed a content-based video retrieval method with orthogonal polynomials [14]. This system identifies keyframes from the input images and uses the colors, textures, angles, and shapes of the visual content. These features are indexed for video retrieval. This method has only been tested on 20 videos, uses only visual attributes, and achieves an accuracy of 69%. This system was quite simple, but it ignored the audio characteristics contained in videos, thus, the performance is not good. Le Wang et al. used an attention-based temporal weighted convolutional neural network (ATW CNN) to identify actions in video [15]. Experimental results on the UCF-101 [16] and HMDB-51 [17] datasets show that the recognition performance of relevant video segments using this model increases significantly from 55.9% to 94.6%. The method eliminated extra information that may cause noise on the image by using appropriate temporal weights to improve the efficiency of feature extraction. Therefore, it requires suitable weights to get better performance and does not consider the processing time.

Many recent studies are looking for new approaches and methods of combining different techniques to provide a scientific basis for future research. Machen Wang et al. solved the problem of multi-person human pose estimation and tracking in videos with an approach consisting of three components, a Clip Tracking Network, a Video Tracking Pipeline, and a SpatialTemporal Merging procedure [18]. The experiments on PoseTrack 2017 and 2018 datasets achieve accuracy from 77.6% to 86.5%. This approach heavily relies on the object images to be identified, and cannot achieve good results with small objects. Recently, Lumin Su et al. introduced ViPNAS, an effective video pose estimation search in both spatial and temporal levels using ResNet-50 and MobileNet v3 as the backbone [19]. Experiments on the COCO2017 and PoseTrack2018 datasets provided high inference speeds without sacrificing accuracy. This work only focuses on image features, ignoring audio features that are also very important in video management. Nils Hjortnaes et al. performed automatic speech recognition with DeepSpeech models for improving the accuracy of Komi language recognition [20]. Their experiments with language models created using KenLM from text materials available online showed significant improvements of over 25% in character error rate and nearly 20% in word error rate. This gave an insight to improve ASR results under low-resource conditions, i.e., the lack of training data. Zerun Feng et al. presented a visual semantic enhanced reasoning network (ViSERN) to exploit reasoning between frame regions using the novel random walk rule-based graph convolutional networks for video-text retrieval [21]. They provided experiments on the MSR-VTT [22] and MSVD [23] datasets. Jianfeng Dong et al. proposed a dual deep encoding network that encodes videos and queries into powerful dense representations of their own [24]. These representations can be transformed to perform sequence-to-sequence cross-modal matching effectively given videos as sequences of frames and queries as sequences of words. The authors provided extensive experiments on four video datasets, i.e., MSR-VTT, TRECVID AVS 2016–2018, VATEX [25], and MPII-MD [26]. Tingtian Li et al. proposed a framework to retrieve background music for fine-grained short videos using the self-attention and cross-modal attention modules to explore the intra- and the inter-relationships

of different modalities, respectively [27]. They built and released two virtual-content video datasets, i.e., HoK400 and CFM400 [27].

Although content-based video recognition has made significant progress in recent years, most research has focused on improving accuracy and ignoring real-time efficiency. There are several works on video analysis in the context of big data. Aftab Alam et al. provided a review on video big data analytics in the cloud and proposed a service-oriented architecture bridging the gap among large-scale video analytics challenges, big data solutions, and cloud computing [28]. Anjali et al. (2019) conducted a survey on multiple object tracking for fast and parallel video processing in MapReduce with the Amazon EC2 Cloud [29]. The results showed that for a large number of videos, the computational speed is faster and the performance is higher when using a fully parallel technique in comparison to a partially parallel technique. The scientific challenge posed for these studies is to solve the best balance between high accuracy and fast response time, especially in large data processing systems. Moreover, a large amount of video data is often required to train and deploy useful machine learning models in industry and entertainment. Smaller enterprises do not have the luxury of accessing enough data for machine learning because of the computational resource limitation. These challenges are critical when developing machine learning algorithms. Several attempts have been made to address the above challenges by using distributed learning techniques such as federated learning over disparate data stores to circumvent the need for centralized data aggregation. This work presents an improved method to train deep neural networks over several data sources in a distributed way, and eliminate the need to centrally aggregate and share the video data. We propose an implementation of content-based video retrieval systems in a parallel and distributed processing environment to improve processing speed and adapt to real-time big data-driven systems. It focuses on the main features, including subtitles, speeches, and objects in video frames. The proposed method allows the training of deep neural networks using video data from multiple nodes in a distributed environment and to secure the representation shared during training. The method was evaluated on existing video data and the performance of this implementation was compared for nine scenarios. This method will pave the way for distributed training of neural networks on privacy-sensitive applications where raw data may not be shared directly or centrally aggregating this data in a data warehouse is not feasible.

3. Background

In this section, we summarize the techniques used in the proposed method, including techniques for the video content extraction such as optical characters, speech, and image objects to query video. In addition, we briefly present the model evaluation metrics in detail below.

3.1. Optical Character Recognition

OCR - Optical Character Recognition [1] is a technique to identify characters on images. Tesseract [30,31] is an open source OCR library, developed by Google. It stands out with advantages such as high accuracy (up to 97%), supporting recognition of many languages, running on many platforms, and running independently or integrating with OpenCV. The identification process will be performed sequentially according to several steps, as shown in Figure 1.

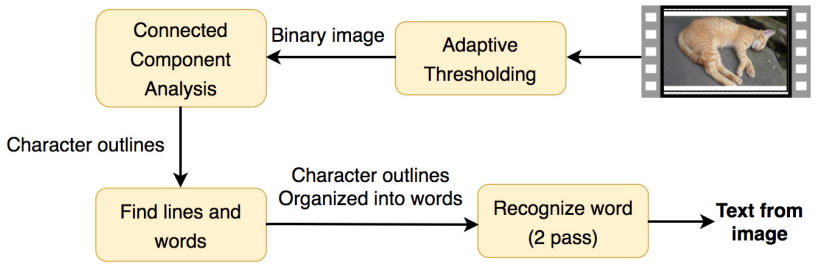


Figure 1. Tesseract recognition algorithm—OCR.

3.2. Speech Recognition

Speech recognition [32] is the task of converting voice signals into text format for search engines. In this study, we use the asynchronous recognition method (REST and gRPC) with Google's SpeechRecognition library. Features are extracted from audio clips and referenced with Google's trained dictionary. The classification is performed and returned in text format (Figure 2).

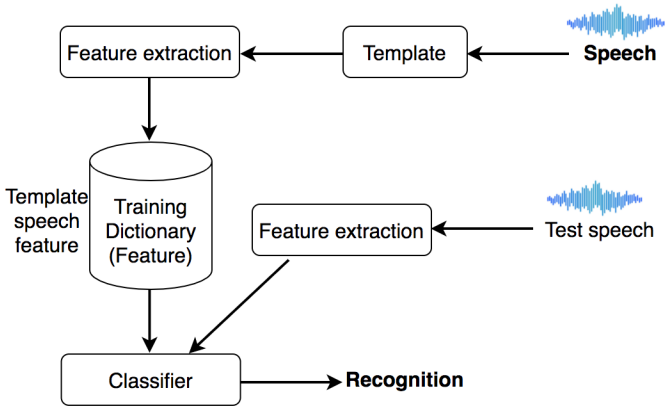


Figure 2. Flowchart of speech recognition process.

3.3. Object Detection and Feature Extraction Using Deep Learning

Deep learning [33] is a subset of machine learning algorithms with high complexity characteristics. The deep learning models are trained with a large labeled dataset and a neural network architecture that learns features directly from the input data without feature extraction. Most deep learning methods use neural network architecture; thus, deep learning models are often called deep neural networks. There are many types of multi-layer neural networks suitable for different types of tasks. Neural networks directly extract features from input images. The features are not pre-trained but they are learned while training on the image datasets. This automatic feature extraction enables highly accurate deep learning models for computer vision tasks such as object classification.

Taking the advantages of deep learning techniques, we build adaptive deep neural networks developed from ResNet, Inception ResNet, MobileNet-V2 for feature extraction, and apply Faster R-CNN and SSD for object recognition. The following is a brief description of these networks.

3.3.1. Deep Neural Networks for Object Detection

Residual Neural Network (ResNet)

The Microsoft Research Team proposed ResNet [34] in 2015. They demonstrated that ResNet is easier to optimize and has higher accuracy than previous models. ResNet is an efficient architecture that won the ImageNet competition by using skip connections. The main challenge in training deep learning models is that accuracy decreases with the depth level of the networks. ResNet converges very quickly and can be trained with hundreds or

thousands of layers. At the same time, ResNet is easy to optimize and can achieve accuracy gains from greatly increased depth, producing better results [35]. Skip connections help to keep information from being lost by connecting the earlier layer to the layer behind and skipping some intermediate layers. Therefore, we propose to apply ResNet for the feature extraction of objects on video frames.

Inception ResNet

To diversify experiments for object features extraction, we propose to use Inception ResNet architecture [36]. It is a model built on the advantages of Inception block and Residual block. Inception ResNet achieves astonishing accuracy with this combination. The complete Inception network consists of many small Inception modules. The idea of the Inception modules is very simple, instead of using a Conv layer with a fixed kernel_size parameter, Inception uses multiple Conv layers at the same time with different kernel_size parameters (1, 3, 5, 7, etc.) and connects the outputs. The input to the model is 299×299 images and the output is a list of class prediction results.

MobileNet-V2

Although the DNN models introduced above are highly accurate, there is a common limitation that they are not suitable for mobile applications or em- bedded systems with low computing capacity. To develop these models for real-time applications, we need an extremely powerful machine configuration (GPU/TPU). A "lighter" model is necessary for embedded systems (Raspberry Pi, Nano pc, etc.) or applications running on smartphones. On the same ImageNet dataset, MobileNet V2 has the same accuracy as other models, such as VGG16 and VGG19, while the number of parameters is only about 3.5 M (about 1/40 of the parameters of VGG16) [37]. Thus, MobileNet-V2 has the advantages of being fast, lightweight, and highly accurate, which is suitable for training with limited datasets. The key point that helps MobileNet models reduce the amount of computation is to apply depthwise separable convolutions.

3.3.2. Network Models for Training and Classification

Faster R-CNN

This is the model architecture that improves both training and detecting speed proposed by Shaoqing Ren et al. at Microsoft Research in 2016 [38]. Faster R-CNN classifies objects and specifies object locations in an image, in which the output is the coordinates of a rectangle and the object inside that rectangle. It has gone through many versions such as R-CNN [39] and Fast R-CNN [40]. Faster R-CNN architecture (Figure 3) is at the pinnacle of the R-CNN family models and achieves the near-best results in object recognition problems. Faster R-CNN uses Region Proposal Network (RPN) instead of selective search algorithms in order to solve the defect of execution time of R-CNN and Fast R-CNN. Faster R-CNN is considered to achieve higher speed and accuracy than it's predecessors. It may not be the simplest or fastest method for object detection, but it is still one of the methods that provides high accuracy. In this study, we propose to use this network model for object detection and classification on video frames to evaluate the experimental results.

Single Shot Detector (SSD)

Recently, a new group of object detection networks has been proposed, in which the region proposal network (RPN) is completely eliminated. It significantly improves processing speed compared to Faster R-CNN but in a completely different way. Faster R-CNN performs two separate phases, one for defining region proposals, the other for objects detection on each region proposal. SSD conducts both tasks in a single phase, predicts bounding boxes and labels while processing images. SSD is a typical example for this group designed to detect objects in real time [41]. This model uses boxes with different scales to identify areas of objects and classify objects. The SSD is conceptually simpler than

the other methods because it eliminates the creation of region proposals thus increasing the processing speed without sacrificing performance.

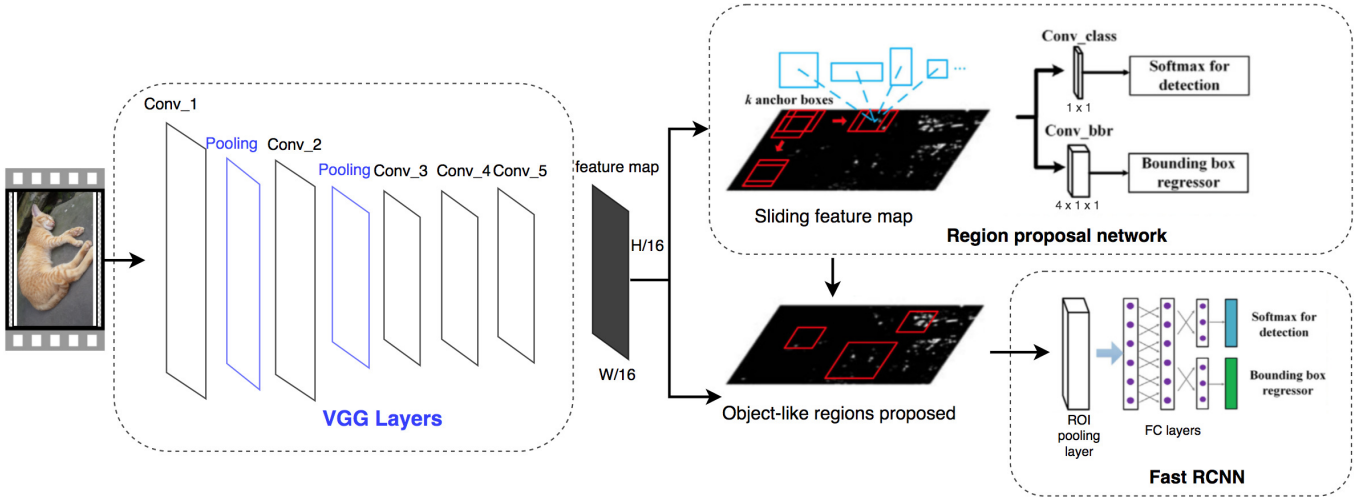


Figure 3. Faster R-CNN ResNet architecture.

3.4. Big Data Processing with Spark

MapReduce [42] has become the most popular model for processing big data on large-scale systems. The scalability of data mining and machine learning algorithms has improved thanks to the MapReduce model. However, the iterative algorithms have not been effectively handled by Hadoop MapReduce because of consecutive accessing files stored in HDFS. To overcome this limitation, Spark is a better choice compared to Hadoop MapReduce since it processes the files in-memory instead of disks, at least 10 times [43–46]. As a result, Spark is used in this work to enable fast and efficient distributed big data processing.

Spark Core is a key component of Spark providing the most basic functions such as task scheduling, memory management, and error recovery. Specifically, it provides an API to define an RDD (resilient distributed dataset), which is a set of resilient elements processed across computing nodes. Figure 4 shows an overview of a distributed computational model with Spark. A Spark job is divided into interdependent stages that can be submitted in parallel to improve the processing throughput. Stages can be shuffle map or result type that consist of multiple tasks. A task is the smallest unit of execution that can be handled by a worker node. The driver node running a Spark job is responsible for scheduling, assigning, and monitoring tasks to worker nodes, which run the actual Spark tasks.

Besides, Spark is compatible with many distributed file storage systems such as HDFS, Cassandra, HBase, and Amazon S3. In this study, input video datasets and extracted features from videos are stored in HDFS to support feature extraction and classification in the Spark environment.

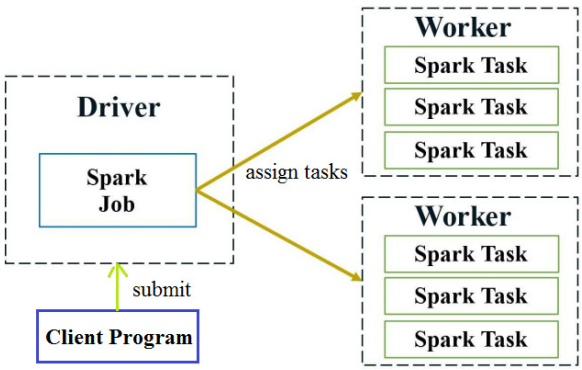


Figure 4. Distributed computing model with Spark.

3.5. Evaluation Metrics

It is necessary to have suitable metrics to evaluate and compare the network models. In this work, we decide to use confusion matrix, AP, mAP, and IoU to evaluate the proposed network models. The details of these metrics are as follows.

3.5.1. Confusion Matrix

A confusion matrix is a table that is often used to evaluate the performance of the classification models (Table 1).

Table 1. Confusion matrix.

Actual Class / Predicted Class	P	N
P	TP	FP
N	FN	TN

Accuracy is a type of measure to evaluate the model by the ratio between the number of correctly classified images to the total number of images. The accuracy is calculated by Equation (1), in which, TP (true positive) is the number of labeled images that are correctly classified; FP (false positive) is the number of labeled images that are misclassified; FN (false negative) is the number of unlabeled images that are correctly classified; TN (true negative) is the number of unlabeled images that are misclassified.

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \tag{1}$$

To evaluate the network models' accuracy, we calculate the Precision and Recall values using Equation (2). High precision means that the accuracy of the predicted cases is high. High recall means that the rate of omission of really positive objects is low.

$$Precision = \frac{TP}{TP + FP}; \quad Recall = \frac{TP}{TP + FN} \tag{2}$$

3.5.2. AP and mAP Metrics

Average precision (AP) is a measurement of accuracy on each class commonly used in classification problems with networks such as SSD and Faster R-CNN. This metric is originally proposed in [47] and it is later used in many studies [48,49]. The AP measurement is calculated using Equation (3). It performs an 11-point interpolation to summarize the shape of the Precision x Recall curve by averaging the precision at a set of 11 evenly spaced recall levels [0, 0.1, 0.2, ..., 1] (Figure 5). The precision $\rho_{interp}(r)$ at each recall level r is interpolated as the maximum precision $\rho(\tilde{r})$ at recall value $\tilde{r}$ greater than or equal to recall level r.

$$AP = \frac{1}{11} \sum_{r \in 0, 0.1, \ldots, 1} \rho_{interp}(r) \qquad with \quad \rho_{interp}(r) = \max_{\tilde{r}: \tilde{r} \geq r} \rho(\tilde{r}) \tag{3}$$

We use mAP (mean average precision) to calculate the average accuracy of all classes to evaluate the general accuracy of the five experimental network models in this study. The mAP measurement is calculated using Equation (4) after obtaining the AP measurement. AP_i is the AP of class i, N is the number of classes.

$$mAP = \frac{1}{N} \sum_{1}^{N} AP_i \tag{4}$$

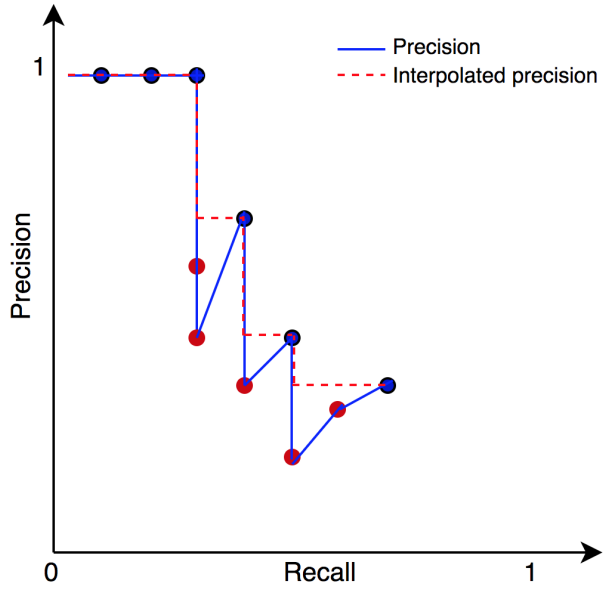


Figure 5. The precision/recall curve.

3.5.3. Intersection over Union (IoU)

The intersection over union (*IoU*) [50] or Jaccard Index is a measure to represent the similarity between the ground truth bounding box and the predicted bounding box of the model. The *IoU* is calculated as Equation (5).

$$IoU_{pred}^{truth} = \frac{truth \cap pred}{truth \cup pred} \tag{5}$$

4. Proposed Method

In this section, we present our proposed approach for extensive feature extraction on big video datasets using deep learning techniques. It includes the distributed and parallel processing model for better processing time, the techniques for content features extraction (speeches, subtitles, and objects), and content indexing for video retrieval. The details of our approach are presented as follows.

4.1. Distributed and Parallel Processing Model for Video Feature Extraction

It is time-consuming work to extract features in large-scale datasets for content-based video retrieval systems. Apache Spark extends the MapReduce model providing the flexibility to persist data records, either in memory, on disk, or both. Spark favors the iterative processes met in machine learning and optimization algorithms. Therefore, we propose the video feature extraction in a distributed and parallel processing model with Spark, as illustrated in Figure 6. This not only saves the extraction processing time but also reduces the training time for deep learning models.

A Spark cluster consists of a manager (master) and workers (slaves). The manager node running Spark Job is responsible for scheduling, assigning, and monitoring tasks to worker nodes, which run the actual Spark tasks. The workers execute tasks and send the status of the tasks to the manager. As a result, the extracted features from videos are stored and indexed in HDFS. It is available for querying by content-based video retrieval systems. The pseudocode algorithm to extract the video content is described in Algorithm 1.

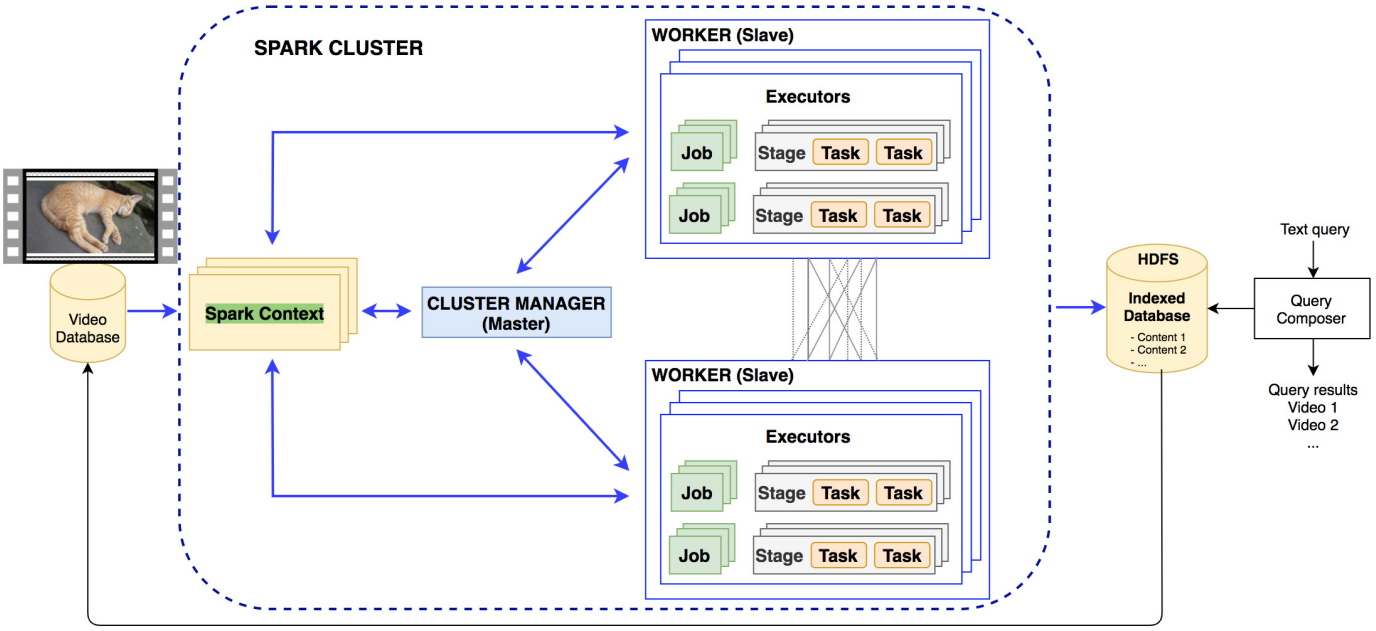


Figure 6. Distributed and parallel processing model with Spark for video feature extraction.

Algorithm 1 Distributed video feature extraction

Input: Video Dataset
Output: Video content features are extracted and indexed
Begin

1: Initialize Spark Context
2: Browse videos in dataset
3: **for each** video in dataset **do**
4: Initialize N worker nodes
5: Split videos into frames, which are then fed into the worker nodes
6: Data pre-processing
7: Extract video features in text format consisting of subtitles, speeches, and objects.
8: Encrypt the extracted features using MD5 algorithm and then store to HDFS.
9: **end for**
10: Indexing the HDFS database containing the video content features

End.

4.2. Feature Extraction

In order to extract features suitable for video retrieval in a big data context, the proposed method uses techniques of OCR, ASR, and deep neural networks implemented in parallel and distributed computing environments. The model of the proposed method is shown in Figure 7. The proposed method for feature extraction includes the following steps: pre-processing, content extraction, shuffling, and sorting. The step of shuffling and sorting occurs simultaneously to summarize the worker's intermediate output.

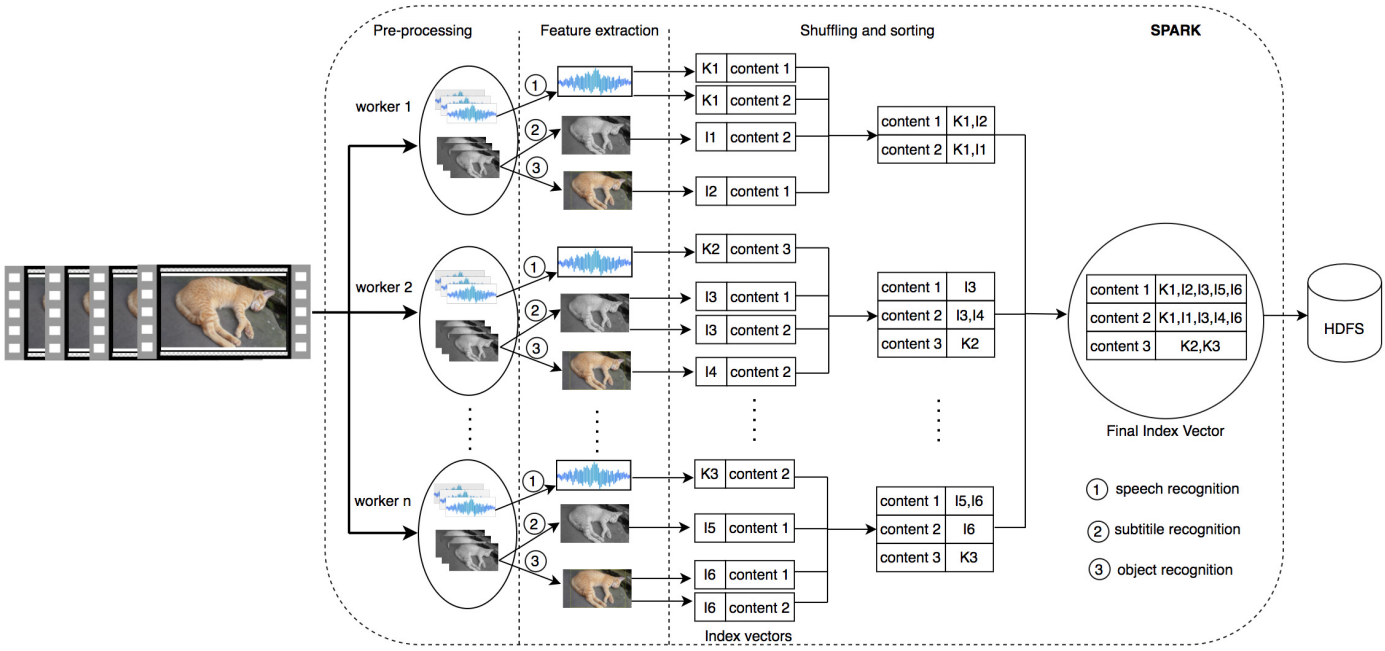


Figure 7. The proposed model of the distributed and parallel computing for feature extraction.

4.2.1. Step 1: Pre-Processing

We focus on extracting the content features consisting of subtitles, speeches, and objects in videos, as shown in Figure 8. Therefore, in this step, we perform the extraction of images and audio clips from the input videos. Each standard input video will have between 25–30 frames per second (fps). These images and audios will be the input data for feature extraction in the next step.

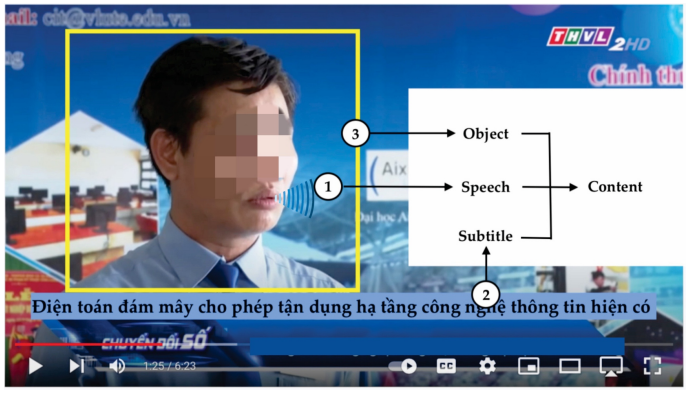


Figure 8. Video segmentation into objects, speeches, and subtitles.

4.2.2. Step 2: Feature Extraction

Video is an electronic medium used to record motion pictures and voices. Extracting content-related features from videos is posing a much bigger challenge than normal text data. After the pre-processing step, we perform a feature extraction of speeches (1) from audio clips, subtitles (2), and objects (3) from image frames in videos as shown in Figure 8. The result of this step is the extracted features in text format stored in HDFS.

Feature Extraction of Speeches

We perform speech recognition using Google's SpeechRecognition library [32]. Audio clips will be converted to wav extension format to match the input requirement of the recognition library. This library supports the recognition of audio clips up to

480 min in length, recognizes many available languages, and supports real-time streaming video. In 2017, Këpuska and Bohouta [51] compared speech recognition systems, such as Sphinx-4, Microsoft Speech API, and Google Speech API by using some audio recordings selected from many sources. The results showed that Sphinx-4 achieved 37% word error rate (WER), Microsoft Speech API achieved 18% WER, and Google API achieved 9% WER. The experimental results of the study stated that the acoustic modeling and language model of Google are superior. Thus, we perform speech recognition using Google's SpeechRecognition library. This library automatically recognizes sounds, determines which objects are voices, identifies language types, and correctly converts them into corresponding texts. With support from machine learning, a lot of features can be updated and improved continuously, meeting all user requirements, fast processing speed, and easy integration. These are the most potential advantages for the big data video system in this study.

Feature Extraction of Subtitles

The image frames will be converted to normalized, histogram equalized, and sharpened images to increase the accuracy of subtitle recognition. Then, subtitles in these images are extracted as text using the Tesseract OCR library [30,31]. This library is easy to retrain with new fonts, supports multi-language recognition with high accuracy, and easily integrates with multiple platforms. In 2022, Cem Dilmegani [52] presented work on the text extraction accuracy of the five most prominent products (Google Cloud Vision, AWS Textract, Tesseract OCR, ABBYY, and Microsoft Azure). Tesseract OCR has high recognition accuracy, just behind Google Cloud Vision and AWS Textract. However, the disadvantage of Google Cloud Vision is the high cost. AWS Textract cannot recognize handwritten text and does not achieve stable performance with complex handwriting. Meanwhile, Tesseract automatically extracts/recognizes text subtitles from images in video, identifies language types, and correctly converts them into corresponding texts. With support from machine learning, a lot of features can be updated and improved continuously, meeting all user requirements, fast processing speed, and easy integration. Therefore, we choose Tesseract OCR to perform the video subtitle extraction because of these advantages, which are suitable for the big data video system in this study.

Feature Extraction of Objects

Deep neural networks (DNN) have been shown to produce highly effective deep learning models in a diverse group of fields. It is popularly used for object classification in images and videos, as presented in Section 3.3. Thus, we leverage the advantages of these networks to develop three adaptive deep neural network models for object detection. Then, the content features are extracted from the detected objects by two network models of Faster R-CNN and SSD. As a result, these content features are stored in HDFS, as shown in Figure 9.

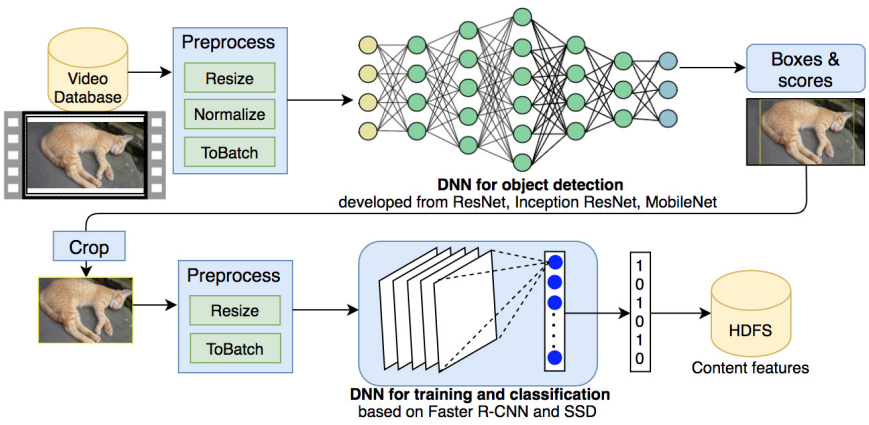


Figure 9. General model for object feature extraction from images in video.

The proposed network models for object detection are developed from the ResNet, Inception ResNet V2, and MobileNet V2 by making some improvements to their layers to adapt for the feature extraction of objects in images. The architectures of the proposed network models for object detection are shown in Figures 10–12. ResNet ensures information integrity by simply learning the residual between input and output. With the advantages of ResNet, we construct model 1 based on ResNet, as shown in Figure 10. We changed the size of the max pooling layer from 3×3 to 2×2 (dotted box). Figure 11 is the proposed network model 2 based on the Inception ResNet V2 with the change of the max pooling size from 3×3 to 2×2. Besides, we also design model 3 inspired by MobileNet V2 changing the size of deepwise, as shown in Figure 12.

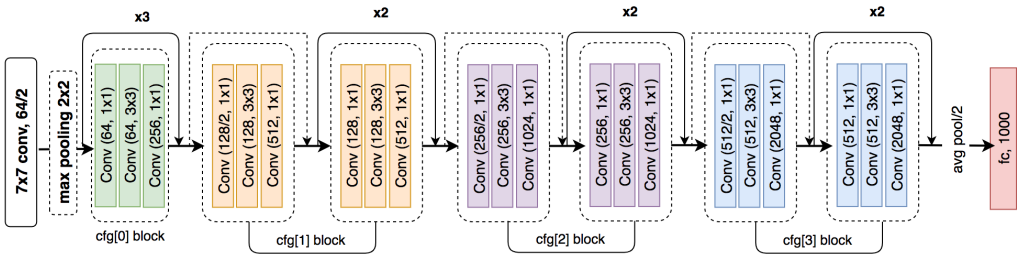


Figure 10. Model 1: Architecture of proposed network 1 for object detection.

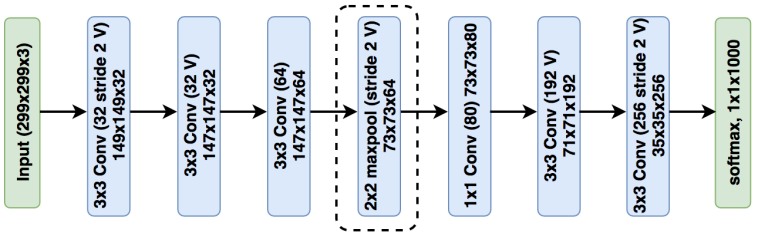


Figure 11. Model 2: Architecture of proposed network 2 for object detection.

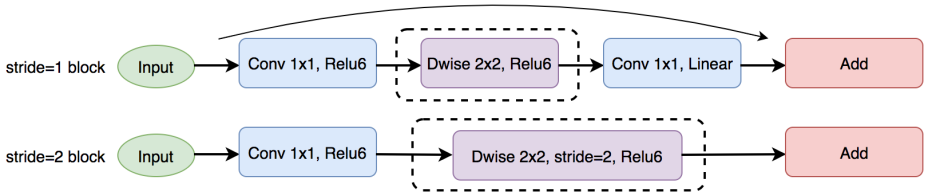


Figure 12. Model 3: Architecture of proposed network 3 for object detection.

Sometimes we do not have a large annotated dataset and do not have the computing resources to train a network model from scratch. In this case, we propose a simple unified solution by taking advantage of the transfer learning approach to train the pre-trained network models. It is useful to reduce the time and space costs of the training and extracting process. The proposed network models have been pre-trained on datasets of ImageNet and COCO. We then use the pre-trained weights and re-train them on our training dataset to fine-tune the parameters of these networks. This leads to faster learning, shorter training time, and no requirement for large training datasets and computing resources.

In order to accommodate large-scale data in a big data context, we design a distributed deep learning model implemented on Spark. Figure 13 describes the structure of a distributed deep neural network to train and extract features from objects in image frames. The Manager node is responsible for the configuration of the cluster, while the worker nodes perform the learning tasks submitted to them through a driver program, along with an initial dataset. In a training parallelization, the Manager is responsible for computing average weights to provide a global average parameter (W) of network parameters, while

the rest of the workers are responsible for training. Each worker obtains the local weights W_i corresponding to its local weights of the network to send updates to the manager node. The same weights W are distributed to all workers when the averaging is executed. After training is complete, we use the same training model with a global average parameter (W) on each worker to query the videos. In an extracting parallelization, the manager is responsible for splitting the image dataset into batches and distributing them along with the global average parameter W to the workers to extract features from videos. The output of this process is that the labels of the objects are automatically extracted representing the content features of videos.

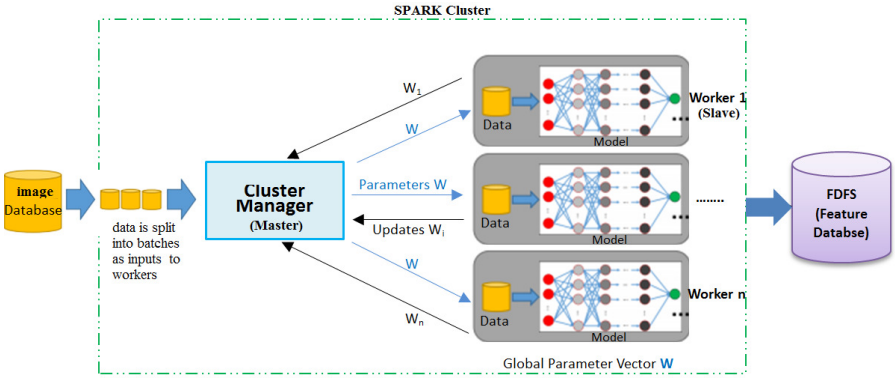


Figure 13. The proposed model for training and extracting object features using a distributed deep learning.

4.3. Content Indexing

Content indexing is the task of arranging documents or keywords formed by the extracted content features to quickly respond to users' queries. To avoid a duplication of the extracted features and ensure the data integrity when stored in databases, these features are encrypted by the MD5 encryption algorithm [53]. The result of the content indexing is a list of the keywords formed by the extracted content features in videos along with links to where they are stored in HDFS.

5. Experiments

In this section, we provide details on the experiments conducted with the proposed approach. The section describes our experimental datasets, the scenarios, and the results obtained after the experiments. These results include experimental results in the training and testing phases. The training results serve as a basis to choose the optimal parameters and create a good model for video querying. The test results are used to evaluate the query results from the trained model.

5.1. Dataset and Installation Environment

The dataset used for our experiments includes videos collected at Vinh Long Radio and Television Station (VLRTS), Vietnam. These videos are randomly taken from categories, such as news and entertainment, to ensure methodological reliability. The dataset is described in Table 2. The original dataset includes 45 videos extracted into 21,505 images and 2140 audio clips. The audio dataset is used for speech extraction while the image dataset for subtitle and object extraction comprises 38 object classes with 38 corresponding labels. The object labels contain the object localization and classification. These 38 classes include people, things, events, or categories that users often search for in VLRTS's programs, suggested by VLRTS's content experts. The image dataset is divided with a ratio of 80:20 for the training dataset and the testing dataset on the proposed neural network models. The quality of the dataset directly affects the accuracy results when training the network models.

Table 2. Experimental datasets.

Dataset	No. Videos	Time (s)	No. Images	No. Audio Clips
10 GB	20	3208	3208	418
20 GB	35	6751	6751	807
30 GB	40	11,825	11,825	1372
60 GB	45	21,505	21,505	2140

The parallel architecture considered to conduct our experimental model is based on TensorFlowOnSpark to enable supporting the proposed distributed DNN models on Apache Spark clusters. The Manager node of the cluster is configured with an Intel Core (TM) i7-3520M CPU@3.6 GHz, 16 GB Ram, and 500 GB disk space. The workers have a configuration with Intel Core i5-9400F CPU@2.9 GHz, 2 GB Ram, and 450 GB disk space. In addition, to compare and evaluate the proposed models, all nodes run the operating system of Ubuntu Linux 16.04, and using Python programming language for scenarios.

5.2. Scenarios

Table 3 describes the experimental scenarios. In the first three scenarios, we only perform speech and subtitle recognition to compare the performance between normal processing and parallel processing on the Spark cluster with a change in the number of worker nodes. In the next scenarios with the same spark environment, we compare and evaluate the proposed neural network models used for object detection and classification.

Table 3. Experimental scenarios.

Sce	Env.	Nodes	Content Extraction Technique		
			Speech	Subtitle	Object (Detector + Extractor and Classifier)
1	Normal	1	Speech-Recognition	Tesseract OCR	no
2	Spark	2	Speech-Recognition	Tesseract OCR	no
3	Spark	3	Speech-Recognition	Tesseract OCR	no
4	Spark	3	Speech-Recognition	Tesseract OCR	Model 1 + Faster R-CNN
5	Spark	3	Speech-Recognition	Tesseract OCR	Model 2 + Faster R-CNN
6	Spark	3	Speech-Recognition	Tesseract OCR	Model 3 + SSD
7	Spark	3	Speech-Recognition	Tesseract OCR	Model 1 + Faster R-CNN
8	Spark	3	Speech-Recognition	Tesseract OCR	Model 2 + Faster R-CNN
9	Spark	3	Speech-Recognition	Tesseract OCR	Model 3 + SSD

5.3. Parameters of the Distributed Deep Learning Networks

For Scenarios 4 to 9, we used the proposed distributed deep learning network models for object detection and classification with the transfer learning approach [54]. To achieve this, we trained the models by labeling 38 object classes, equivalent to 38 labels for the training image dataset. We used the pre-trained parameters of the trained models on the common datasets such as ImageNet and COCO. Then, we re-trained the proposed models on our training dataset to fine-turn the model parameters for our use case. This helps solve the problem of the small training dataset and fast training time, while keeping the advantage of the deep neural network models. Moreover, the proposed neural network models are conducted in parallel and distributed computing on Spark, as presented in Section Feature extraction of objects. Table 4 shows the training parameters that we used for Scenarios 4 to 9.

Table 4. Parameters for Scenarios 4 to 9.

Sce	Learning Rate	Batch Size	No. Classes	Scales	Max Pooling Size	Epoches	IoU
4	0.0003	1	38	[0.25, 0.5, 1.0, 2.0]	3×3	50,000	0.5
5	0.0003	1	38	[0.25, 0.5, 1.0, 2.0]	3×3	50,000	0.5
6	0.0003	24	38	[0.3, 0.95]	3×3	50,000	0.5
7	0.0003	1	38	[0.25, 0.5, 1.0, 2.0]	2×2	100,000	0.5
8	0.0003	1	38	[0.25, 0.5, 1.0, 2.0]	2×2	100,000	0.5
9	0.0003	24	38	[0.3, 0.95]	2×2	100,000	0.5

5.4. Experimental Results

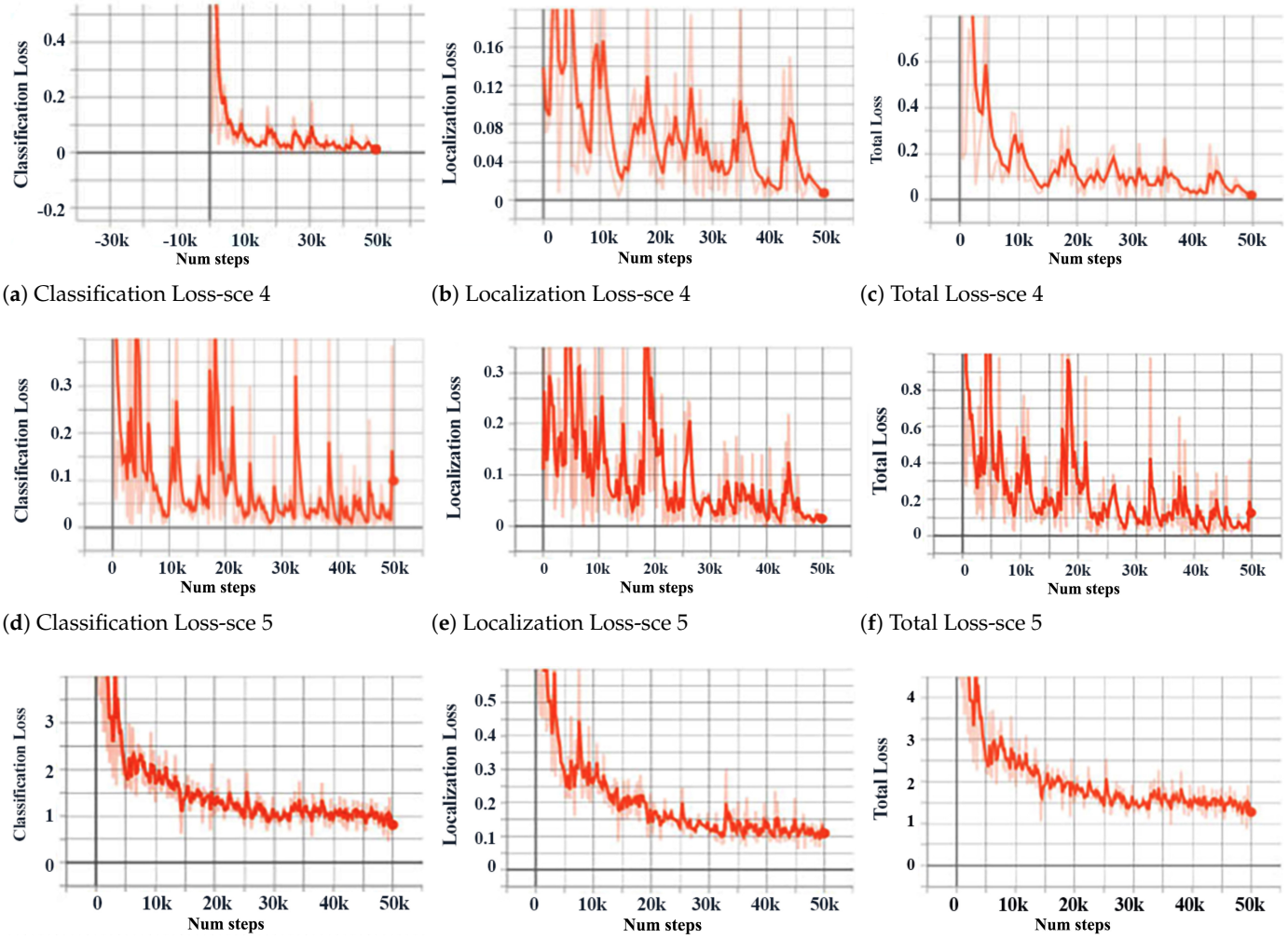
5.4.1. Training Results

The goal of the training process is to seek a set of scenario weight parameters to reduce the scenario error in the next evaluation. Thus, the loss function is used to estimate the error of the scenarios and update the parameters. In the training phase, we seek the optimal parameters of the scenarios by calculating the lowest errors to make a decision to stop training. In this section, we present the training results consisting of the Loss_value measure and the training time for Scenarios 4–9. The remaining scenarios (Scenarios 1–3) do not go through the training phase.

Loss_Value Measure

In order to calculate and minimize the scenario error, the proposed neural networks are trained by an optimization process base on the loss values. Loss value measures the performance of the models. If the model errors are high, the loss will be high (the model does not do a good job) and vice versa. In Scenarios 4–9, we estimate the values of the classification loss, localization loss, and total loss to evaluate the scenario error. Classification loss measures the predictive inaccuracy of classification models. The localization loss is an error function used to calculate the error value for the predicted boundary box, including the coordinates of the center, width, and height of relative to ground truth box from the models' training data. The total loss is the sum of the two loss functions. Figures 14 and 15 show the histograms of the loss function for Scenarios 4 to 9. From the histograms of the loss function, we can see that the total loss is minimal when the number of training steps increases up to 50,000 for Scenarios 4–6, as represented in Figure 14c,f,i, respectively. Scenarios 4, 5, and 6 had total losses of 0.1, 0.2, and 1.5, respectively. It means that the error of Scenario 4 is the lowest in Scenarios 4–6.

Figure 15c,f,i show the histograms of the total loss for Scenarios 7, 8, and 9, respectively, after 50,000 training steps. Obviously, the curve of the total loss function in scenario 8 (Figure 15f) is rapidly decreasing to the lowest, less than 0.05, compared to the remaining scenarios and it is stable at the lowest after 50,000 training steps. It means that the error of Scenario 8 is the lowest compared to the remaining scenarios with only 5%.



(**a**) Classification Loss-sce 4 (**b**) Localization Loss-sce 4 (**c**) Total Loss-sce 4

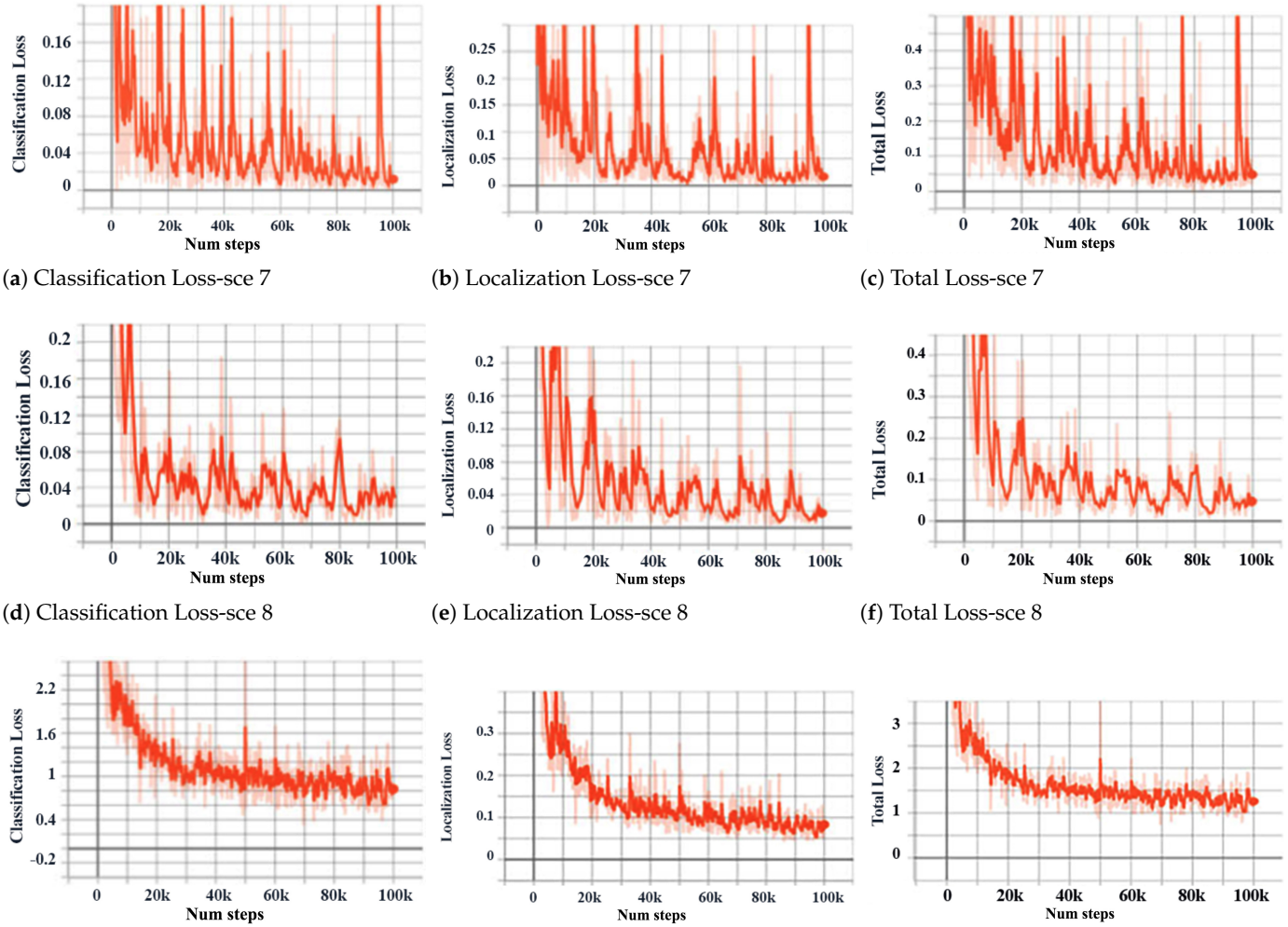
(**d**) Classification Loss-sce 5 (**e**) Localization Loss-sce 5 (**f**) Total Loss-sce 5

(**g**) Classification Loss-sce 6 (**h**) Localization Loss-sce 6 (**i**) Total Loss-sce 6

Figure 14. Histograms of the loss values for Scenarios 4, 5, and 6.

Training Run-Time

All three scenarios (1–3) had no training so training time is 0. The training time for the remaining six scenarios is illustrated in Figure 16. The training time for Scenarios 4 and 7 was 7.2 h and 16.17 h, respectively. For Scenarios 5 and 8, the training time was 6.68 h and 15.68 h, respectively. It was 5.45 h and 8.63 h for Scenarios 6 and 9, respectively. Scenario 7 had the longest training time compared to other scenarios. Scenario 6 was the fastest to train. Although Scenario 8 had a longer training time than Scenarios 4–6, and 9, and its training time was faster than Scenario 7, its error was the lowest. Therefore, we can conclude that Scenario 8 is one of the most suitable scenarios for object feature extraction from images for content-based video retrieval.



(**a**) Classification Loss-sce 7 (**b**) Localization Loss-sce 7 (**c**) Total Loss-sce 7

(**d**) Classification Loss-sce 8 (**e**) Localization Loss-sce 8 (**f**) Total Loss-sce 8

(**g**) Classification Loss-sce 9 (**h**) Localization Loss-sce 9 (**i**) Total Loss-sce 9

Figure 15. Histograms of the loss values for Scenarios 7, 8, and 9.

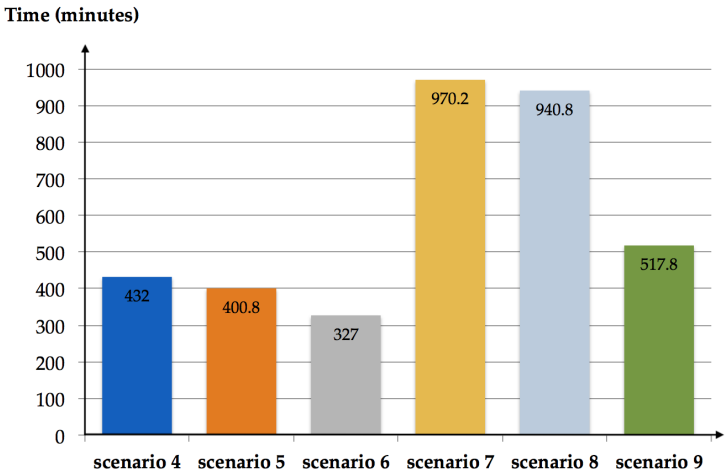


Figure 16. Training time in minutes for Scenarios 4 to 9.

5.4.2. Test Results

In order to evaluate the proposed scenarios, we determined the measures of AP, mAP, and the execution time for feature extraction. The following is an analysis of the test results across nine scenarios.

Average Accuracy and Execution Time for Scenarios 1–3

We tested the first three Scenarios 1–3 on four datasets, as described in Table 2. In a parallel computing environment, the accuracy of these scenarios does not change with an increasing number of worker nodes but changes on these four datasets when extracting features of the speech and subtitle. Figure 17 represents the average accuracy of the scenarios corresponding to each dataset. The accuracy of speech recognition was higher than that of subtitle recognition. The accuracy of subtitle recognition reached from 78.24% to 82.17%, while the accuracy of speech recognition reached from 90.1% to 91% on the four datasets. The subtitle and speech recognition achieved an average accuracy of 85%.

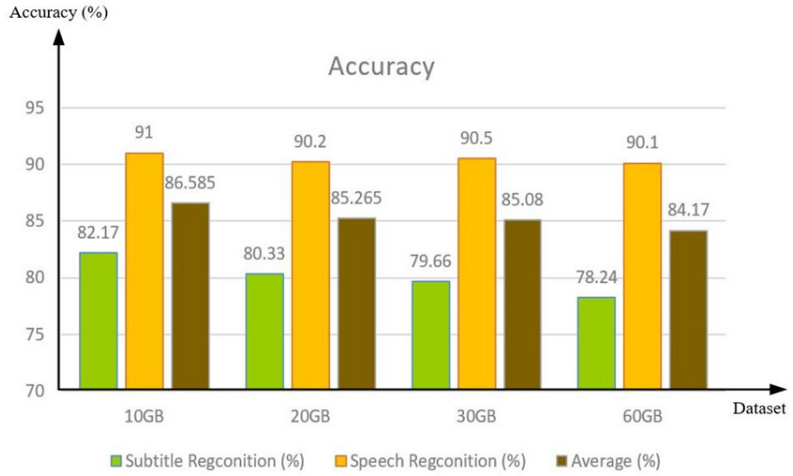


Figure 17. Average accuracy for three Scenarios 1–3.

Meanwhile, the execution time of the three Scenarios 1–3 rapidly decreased in the parallel and distributed computing environment as the number of worker nodes increased, as shown in Figure 18. We compare the execution time of these scenarios on four datasets corresponding to changing the number of nodes from 1–3. The parallel execution of Scenarios 1–3, with three worker nodes, was shortened by 50% time compared to normal execution. In particular, for three worker nodes, the parallel execution was shortened by 51.4% and 59% time compared to the normal execution for the corresponding dataset sizes of the 10 GB and 60 GB. It is clear that the parallel and distributed processing on Spark has a fast processing time as the number of nodes increases for large-scale datasets.

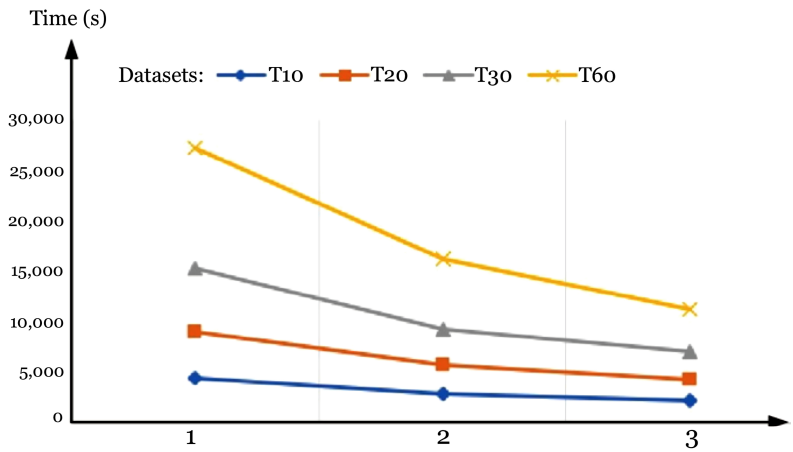


Figure 18. Execution time in seconds for Scenarios 1–3.

Measures of AP and mAP for Scenarios 4–9

We evaluate Scenarios 4 to 9 by calculating AP, mAP, and run-time of the feature extraction. Figures 19 and 20 describe the extraction accuracy of Scenarios 4–9 with two measures of AP and mAP. In Figure 19, Scenario 8 has the most stable extraction results compared to the remaining scenarios for 38 classes. Figure 20 shows that Scenarios 4 and 8 had the highest mAP measure with 0.95 and 0.96, respectively. Scenarios 6 and 9 achieve the lowest mAP measures of 0.86 and 0.88, respectively. Scenarios 5 and 7 show the mAP measures of 0.93 and 0.94, respectively. Scenario 8 gives the results of feature extraction with the highest accuracy compared to the remaining scenarios.

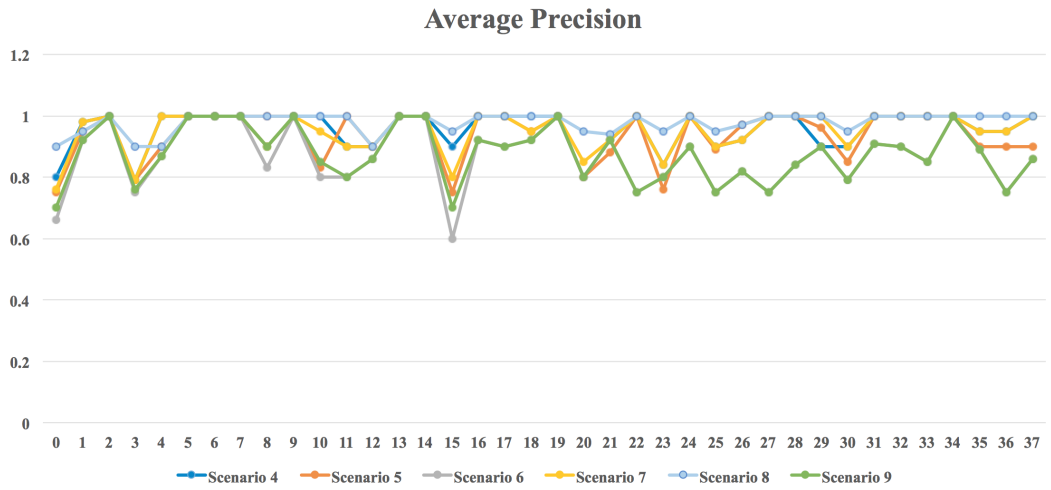


Figure 19. AP measure of each class for Scenarios 4 to 9.

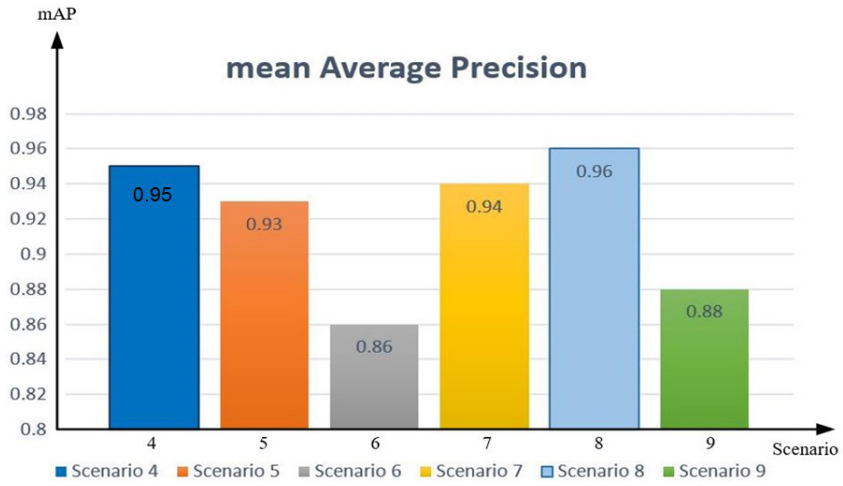
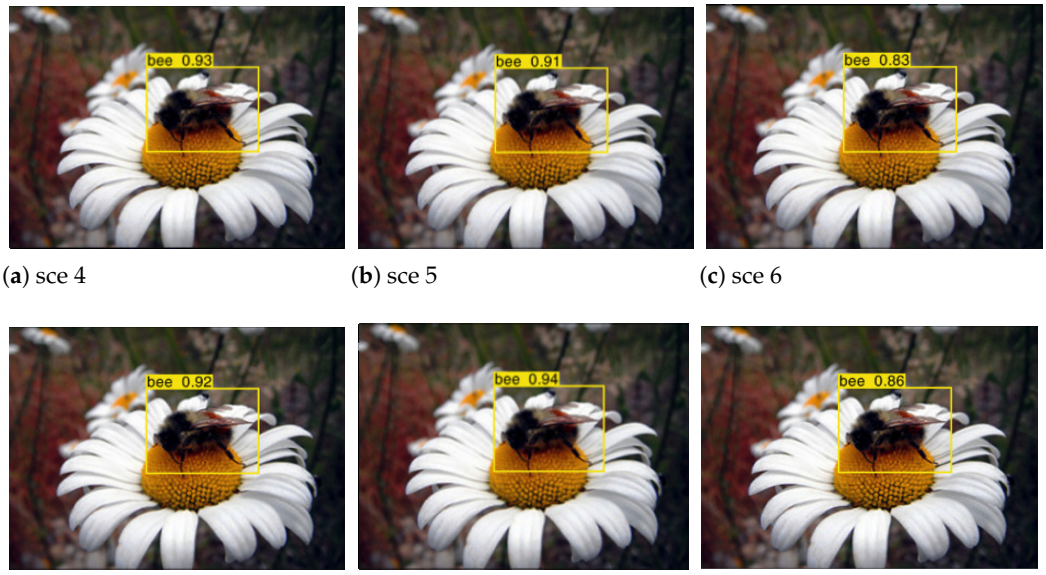


Figure 20. Measure of mAP for Scenarios 4 to 9.

Illustrative Results of the Object Recognition

Some illustrative results of the object recognition for Scenarios 4–9 are presented in Figure 21. We can see that Scenario 8 detects the bee object with the highest accuracy of 94% (Figure 21e). Meanwhile, the bee object detection in Scenarios 6 and 9 has the lowest accuracy of 83% and 86%, respectively.



(**a**) sce 4 (**b**) sce 5 (**c**) sce 6

(**d**) sce 7 (**e**) sce 8 (**f**) sce 9

Figure 21. Illustration of object recognition for Scenarios 4–9.

6. Comparison and Discussion

In our work, we extract the video contents for retrieval using a combination of three features (subtitles, speeches, and object labels). The experimental results of Scenarios 1 to 9 show that the proposed method for content-based video retrieval achieves a high accuracy from 85% to 96% (Figure 22). In particular, Scenario 8 obtained the highest accuracy of 96% for feature extraction of videos using the proposed distributed deep learning model on Spark. Figures 22 and 23 represent the summary of the scenarios' performance. The experimental results show that the processing time in Spark is shortened by 50% without reducing the accuracy when the dataset is increased by six times compared with a normal computing environment (Scenario 3 versus Scenario 1). Besides, Scenario 8 achieves the highest accuracy compared to the remaining scenarios. The average execution time for Scenario 3 is the lowest because it just extracts the speech and subtitle features.

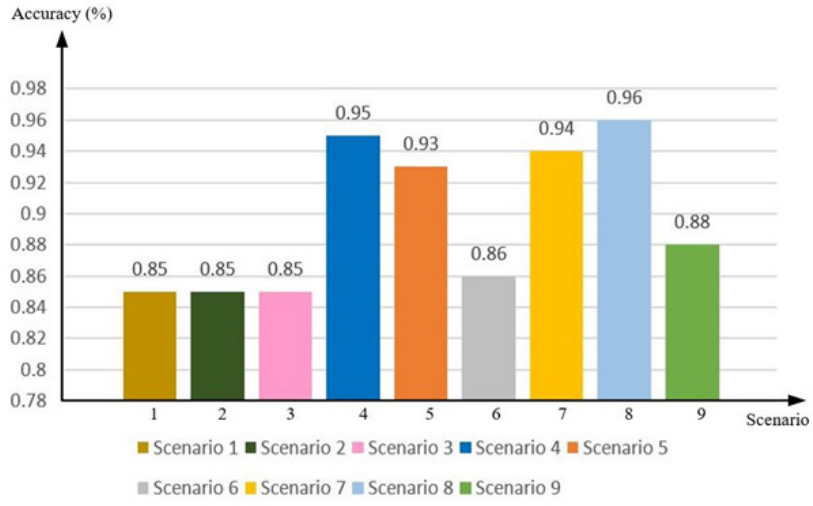


Figure 22. Average accuracy for Scenarios 1 to 9.

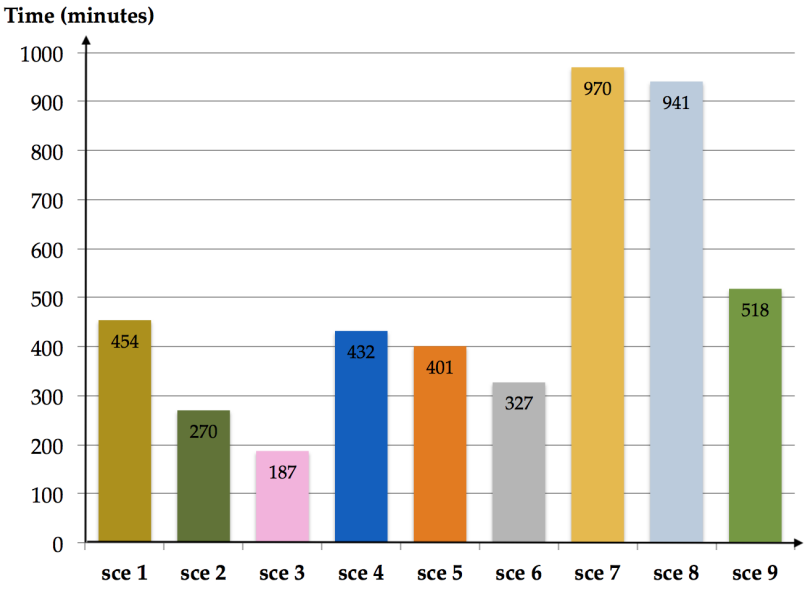


Figure 23. Execution time in seconds for Scenarios 1 to 9.

Using the proposed parallel model on Spark, the execution time is improved faster than other regular models as described in the result analysis of Scenarios 1–3. Figure 23 shows a comparison of the run-time of Scenarios 1 to 9. Scenario 7 consumes the most time for feature extraction while Scenario 8 takes a relatively long time but its average accuracy (AP and mAP measures) is the highest. Through experimental scenarios, it emphasizes the advantages of the proposed method using the distributed deep learning model on Spark. It is suitable for large-scale datasets providing reasonable training time with high accuracy. We can conclude that Scenarios 4 and 8 give better results than the remaining scenarios.

In addition, some experimental comparisons between the proposed methods and the preceding methods are also made as illustrated in Table 5. We perform subject, object, and speech recognition on several open datasets such as TextCaps (https://textvqa.org/textcaps/dataset/, accessed on 20 May 2022), AVSpeech (http://festvox.org/cmu_wilderness/VIEVOV/index.html, accessed on 20 May 2022), and Fruits-360 (https://public.roboflow.com/classification/fruits-dataset, accessed on 20 May 2022). The results show that our proposed method achieves higher accuracy than the previous methods from 1.4% to 2%. We also provide a combination of two video feature recognitions on the video dataset of Vinh Long Radio and Television Station. Our method achieved a 1.4% to 4.2% higher accuracy than the previous studies. It shows that our proposed method not only improves the processing time with a distributed deep learning environment but also increases the recognition accuracy. We incorporate more video features than previous studies to reduce the possibility of missing information for content-based video retrieval systems.

Table 5. Comparison of content-based video retrieval methods.

Video Features	Method	Accuracy	Time (s)	Dataset
Subject	Use a high-level concept word detector that can be integrated with any video-to-language models [8]	67%	10.090	**TextCap**
	Use a Visual Semantic Enhanced Reasoning Network (ViSERN) to exploit reasoning between frame regions [21]	75.9%	7457	
	Use Dual Encoding, with Transformer features included [24]	76.2%	8342	
	Our proposed method	**78.24%**	**7253**	

Table 5. *Cont.*

Video Features	Method	Accuracy	Time (s)	Dataset
Speech	Use the query by example video retrieval [9]	84.35%	7592	**CMU Wilderness Multilingual Speech**
	Use CNN-based multimodal networks [13]	69.78–88.56%	6602	
	Use DeepSpeech [20]	81.6%	8253	
	Our proposed method	**90.1%**	**6272**	
Object	Use orthogonal polynomials [14]	69%	1047	**Fruits-360**
	Use attention-based temporal weighted CNN [15]	55.9–94.6%	1863	
	Use a Clip Tracking Network, a Video Tracking Pipeline and a Spatial Temporal Merging [18]	86.5%	88	
	Use a novel neural architecture search (NAS) method, termed ViP-NAS [19]	81.7%	72	
	Our proposed method	**96%**	**91**	
Object + Speech	Use Supervised Deep Canonical Correlation Analysis (S-DCCA) [4]	85.28%	9065	**THVL Dataset**
	Use the self-attention (SA) and the cross-modal attention (CMA) modules [27]	91.6%	6,907	
	Use a Clip Tracking Network, a Video Tracking Pipeline and a Spatial Temporal Merging [18]	89.61%	7343	
	Our proposed method	**93%**	**4879**	
Subject + Object	Use a Visual Semantic Enhanced Reasoning Network (ViSERN) to exploit reasoning between frame regions [21]	81.3%	8263	
	Use Dual Encoding, with Transformer features included [24]	85.9%	8296	
	Our proposed method	**87%**	**7187**	
Subtitle + Speech	Use DeepSpeech [20]	79.8%	16,188	
	Our proposed method	**84%**	**10,672**	

7. Conclusions

In this study, we proposed an efficient method with nine scenarios of feature extraction for indexing and retrieving the content-based videos in a big data context. We focus on the main features comprising subtitles, speeches, and objects, which form the video content. The proposed method with three first scenarios extracts the features with the number of nodes increasing from 1 to 3, respectively. In a parallel and distributed environment on Spark, the proposed method with six remaining scenarios uses a combination of three techniques, which are automatic speech recognition, subtitle recognition, and object identification with the distributed deep learning approach. With the scenarios of the proposed method, we extract the features from the video database to store and manage for indexing and querying in content-based video retrieval systems. For object identification to extract features, we construct three deep neural network models developed from Faster R-CNN ResNet, Faster R-CNN Inception ResNet V2, and Single Shot Detector MobileNet V2. To train these networks, we use the transfer learning approach and implement the distributed and parallel computing environment on Spark. We leverage the advantage of transfer learning to pre-train the proposed networks on datasets of ImageNet and COCO, and then train them on our datasets. This helps to solve the problem of small training datasets and gives fast training time. Moreover, in a parallel computing environment on Spark, it allows for reducing the costs of time, space, and computing resources in the feature extraction from large video datasets. We also have a comparison of normal computing-based video extraction and distributed video extraction. Some experimental comparisons

between the proposed method and the preceding methods are also made to discuss the advantage of the proposed method.

Currently, multimedia data is collected from various sources with many different formats. Processing, transporting, and storing these data, especially videos, are costly. A multimedia information systems need to adapt to the growing big data environment. These challenges form the basis for our proposed approach to content-based video retrieval in a big data processing environment. Many recent studies focus on voice-based or object-based video queries. The novelty in our approach is a combination of the voice, subtitle, and image objects for querying videos with more detailed information. This approach towards building machine learning models that run distributed across multiple computing nodes that can take advantage of distributed storage and computation. Spark's in-memory processing capabilities significantly reduce the cost of data transmission across the network and increase processing speed.

The experimental results demonstrate that our proposed method using distributed deep learning on Spark achieves the accuracy of 96% and the processing time is shortened by 50% compared to the other methods without Spark. The proposed method can extract strong video features suitable for big data-driven video retrieval systems. This work can be used as a scientific basis for relevant studies on video processing for real-time big data video retrieval systems. In future directions, we will study methods for extracting associative features related to video content and compare them when deployed across multiple clusters.

Author Contributions: Conceptualization, A.-C.P. and T.-C.P.; methodology, A.-C.P. and T.-C.P.; software, A.-C.P. and T.-C.P.; validation, T.-N.T., H.-P.C. and T.-C.P.; formal analysis, A.-C.P.; investigation, A.-C.P. and T.-C.P.; resources, A.-C.P., H.-P.C. and T.-C.P.; data curation, T.-N.T.; writing—original draft preparation, A.-C.P.; writing—review and editing, T.-N.T., H.-P.C. and T.-C.P.; visualization, T.-N.T.; supervision, T.-C.P. and H.-P.C.; project administration, A.-C.P. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data are available on request by contacting the corresponding author.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Yang, H.; Siebert, M.; Luhne, P.; Sack, H.; Meinel, C. Lecture video indexing and analysis using video ocr technology. In Proceedings of the 2011 Seventh International Conference on Signal Image Technology & Internet-Based Systems, Dijon, France, 28 November–1 December 2011; pp. 54–61.
2. El Ouadrhiri, A.A.; Saoudi, E.M.; Andaloussi, S.J.; Ouchetto, O.; Sekkaki, A. Content based video retrieval based on bounded coordinate of motion histogram. In Proceedings of the 2017 4th International Conference on Control, Decision and Information Technologies (CoDIT), Barcelona, Spain, 5–7 April 2017; pp. 0573–0578.
3. Accattoli, S.; Sernani, P.; Falcionelli, N.; Mekuria, D.N.; Dragoni, A.F. Violence detection in videos by combining 3D convolutional neural networks and support vector machines. *Appl. Artif. Intell.* **2020**, *34*, 329–344. [CrossRef]
4. Zeng, D.; Yu, Y.; Oyama, K. Audio-visual embedding for cross-modal music video retrieval through supervised deep CCA. In Proceedings of the 2018 IEEE International Symposium on Multimedia (ISM), Taichung, Taiwan, 10–12 December 2018; pp. 143–150.
5. Huang, J.; Rathod, V.; Sun, C.; Zhu, M.; Korattikara, A.; Fathi, A.; Fischer, I.; Wojna, Z.; Song, Y.; Guadarrama, S.; et al. Speed/accuracy trade-offs for modern convolutional object detectors. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 7310–7311.
6. Borthakur, D. HDFS architecture guide. *Hadoop Apache Proj.* **2008**, *53*, 2.
7. Phan, A.C.; Phan, T.C.; Trieu, T.N. A Systematic Approach to Healthcare Knowledge Management Systems in the Era of Big Data and Artificial Intelligence. *Appl. Sci.* **2022**, *12*, 4455. [CrossRef]

8. Yu, Y.; Ko, H.; Choi, J.; Kim, G. End-to-End Concept Word Detection for Video Captioning, Retrieval, and Question Answering. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 3261–3269.
9. Avgoustinakis, P.; Kordopatis-Zilos, G.; Papadopoulos, S.; Symeonidis, A.L.; Kompatsiaris, I. Audio-based Near-Duplicate Video Retrieval with Audio Similarity Learning. In Proceedings of the 2020 25th International Conference on Pattern Recognition (ICPR); Milan, Italy, 10–15 January 2021; pp. 5828–5835.
10. Kordopatis-Zilos, G.; Papadopoulos, S.; Patras, I.; Kompatsiaris, I. FIVR: Fine-Grained Incident Video Retrieval. *IEEE Trans. Multimed.* **2019**, *21*, 2638–2652. doi: 10.1109/TMM.2019.2905741. [CrossRef]
11. Jiang, Q.Y.; He, Y.; Li, G.; Lin, J.; Li, L.; Li, W.J. SVD: A large-scale short video dataset for near-duplicate video retrieval. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Korea, 27 October–2 November 2019; pp. 5281–5289.
12. Revaud, J.; Douze, M.; Schmid, C.; Jégou, H. Event retrieval in large video collections with circulant temporal encoding. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Portland, OR, USA, 23–28 June 2013; pp. 2459–2466.
13. Pandeya, Y.R.; Lee, J. Deep learning-based late fusion of multimodal information for emotion classification of music video. *Multimed. Tools Appl.* **2021**, *80*, 2887–2905. [CrossRef]
14. Braveen, M. Content Based Video Retrieval with Orthogonal Polynomials. Ph.D. Thesis, Anna University, Chennai, India, 2018.
15. Wang, L.; Zang, J.; Zhang, Q.; Niu, Z.; Hua, G.; Zheng, N. Action recognition by an attention-aware temporal weighted convolutional neural network. *Sensors* **2018**, *18*, 1979. [CrossRef] [PubMed]
16. Soomro, K.; Zamir, A.R.; Shah, M. UCF101: A dataset of 101 human actions classes from videos in the wild. *arXiv* **2012**, arXiv:1212.0402.
17. Kuehne, H.; Jhuang, H.; Garrote, E.; Poggio, T.; Serre, T. HMDB: A large video database for human motion recognition. In Proceedings of the 2011 International Conference on Computer Vision, Barcelona, Spain, 6–13 November 2011; pp. 2556–2563.
18. Wang, M.; Tighe, J.; Modolo, D. Combining detection and tracking for human pose estimation in videos. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 11088–11096.
19. Xu, L.; Guan, Y.; Jin, S.; Liu, W.; Qian, C.; Luo, P.; Ouyang, W.; Wang, X. Vipnas: Efficient video pose estimation via neural architecture search. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 16072–16081.
20. Hjortnaes, N.; Arkhangelskiy, T.; Partanen, N.; Rießler, M.; Tyers, F.M. Improving the language model for low-resource ASR with online text corpora. In Proceedings of the 1st joint SLTU and CCURL Workshop (SLTU-CCURL 2020), European Language Resources Association (ELRA), Marseille, France, 11–16 May 2020.
21. Feng, Z.; Zeng, Z.; Guo, C.; Li, Z. Exploiting Visual Semantic Reasoning for Video-Text Retrieval. In Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, Yokohama, Japan, 7–15 January 2021; IJCAI'20.
22. Xu, J.; Mei, T.; Yao, T.; Rui, Y. Msr-vtt: A large video description dataset for bridging video and language. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 5288–5296.
23. Chen, D.; Dolan, W.B. Collecting highly parallel data for paraphrase evaluation. In Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Portland, OR, USA, 19–24 June 2011; pp. 190–200.
24. Dong, J.; Li, X.; Xu, C.; Yang, X.; Yang, G.; Wang, X.; Wang, M. Dual Encoding for Video Retrieval by Text. *IEEE Trans. Pattern Anal. Mach. Intell.* **2021**, 44, 4065–4080. [CrossRef] [PubMed]
25. Wang, X.; Wu, J.; Chen, J.; Li, L.; Wang, Y.F.; Wang, W.Y. Vatex: A large-scale, high-quality multilingual dataset for video-and-language research. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Korea, 27 October–2 November 2019; pp. 4581–4591.
26. Andriluka, M.; Pishchulin, L.; Gehler, P.; Schiele, B. 2d human pose estimation: New benchmark and state of the art analysis. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2014; pp. 3686–3693.
27. Li, T.; Sun, Z.; Zhang, H.; Li, J.; Wu, Z.; Zhan, H.; Yu, Y.; Shi, H. Deep Music Retrieval for Fine-Grained Videos by Exploiting Cross-Modal-Encoded Voice-Overs. In Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual, 11–15 July 2021; Association for Computing Machinery: New York, NY, USA, 2021; pp. 1880–1884.
28. Alam, A.; Ullah, I.; Lee, Y.K. Video Big Data Analytics in the Cloud: A Reference Architecture, Survey, Opportunities, and Open Research Issues. *IEEE Access* **2020**, *8*, 152377–152422. [CrossRef]
29. Parihar, A.; Nagarkar, P.; Bhosale, V.; Desale, K. Survey on Multiple Objects Tracking in Video Analytics. *Int. J. Comput. Appl.* **2019**, *181*, 5–9. [CrossRef]
30. Boiangiu, C.A.; Ioanitescu, R.; Dragomir, R.C. Voting-Based Ocr System. *J. Inf. Syst. Oper. Manag.* **2016**, *10*, 470–486.
31. Kay, A. Tesseract: An Open-Source Optical Character Recognition Engine. *Linux J.* **2007**, *2007*, 2.
32. Chandra, E.; Akila, A. An overview of speech recognition and speech synthesis algorithms. *Int. J. Comput. Technol. Appl.* **2012**, *3*, 1426–1430.
33. Shinde, P.P.; Shah, S. A review of machine learning and deep learning applications. In Proceedings of the 2018 Fourth International Conference on Computing Communication Control and Automation (ICCUBEA), Pune, India, 16–18 August 2018; pp. 1–6.

34. Szegedy, C.; Ioffe, S.; Vanhoucke, V.; Alemi, A. Inception-v4, inception-resnet and the impact of residual connections on learning. In Proceedings of the AAAI Conference on Artificial Intelligence, Phoenix, AZ, USA, 12–17 February 2017.
35. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
36. Nazir, U.; Khurshid, N.; Ahmed Bhimra, M.; Taj, M. Tiny-Inception-ResNet-v2: Using deep learning for eliminating bonded labors of brick kilns in South Asia. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, Long Beach, CA, USA, 16–17 June 2019; pp. 39–43.
37. Sandler, M.; Howard, A.; Zhu, M.; Zhmoginov, A.; Chen, L.C. Mobilenetv2: Inverted residuals and linear bottlenecks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 4510–4520.
38. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards real-time object detection with region proposal networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **2016**, *39*, 1137–1149. [CrossRef] [PubMed]
39. Girshick, R.; Donahue, J.; Darrell, T.; Malik, J. Rich feature hierarchies for accurate object detection and semantic segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2014; pp. 580–587.
40. Girshick, R. Fast r-cnn. In Proceedings of the IEEE International Conference on Computer Vision, Santiago, Chile, 7–13 December 2015; pp. 1440–1448.
41. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.Y.; Berg, A.C. Ssd: Single shot multibox detector. In Proceedings of the European Conference on Computer, Amsterdam, The Netherlands, 11–14 October 2016; Springer: Berlin/Heidelberg, Germany, 2016; pp. 21–37.
42. Dean, J.; Ghemawat, S. MapReduce: Simplified data processing on large clusters. *Commun. ACM* **2008**, *51*, 107–113. [CrossRef]
43. Zaharia, M.; Chowdhury, M.; Franklin, M.J.; Shenker, S.; Stoica, I. Spark: Cluster computing with working sets. In Proceedings of the 2nd USENIX Workshop on Hot Topics in Cloud Computing (HotCloud 10), Boston, MA, USA, 22 June 2010.
44. Singh, D.; Reddy, C.K. A survey on platforms for big data analytics. *J. Big Data* **2015**, *2*, 1–20. [CrossRef] [PubMed]
45. Aziz, K.; Zaidouni, D.; Bellafkih, M. Real-time data analysis using Spark and Hadoop. In Proceedings of the 2018 4th international conference on optimization and applications (ICOA), Mohammedia, Morocco, 26–27 April 2018; pp. 1–6.
46. Zecevic, P.; Bonaci, M. *Spark in Action*, 1st ed.; Manning Publications Co.: Shelter Island, NY, USA, 2016.
47. Salton, G.; McGill, M.J. *Introduction to Modern Information Retrieval*; Mcgraw-Hill: New York, NY, USA, 1983.
48. Everingham, M.; Van Gool, L.; Williams, C.K.; Winn, J.; Zisserman, A. The pascal visual object classes (voc) challenge. *Int. J. Comput. Vis.* **2010**, *88*, 303–338. [CrossRef]
49. Simonelli, A.; Bulo, S.R.; Porzi, L.; López-Antequera, M.; Kontschieder, P. Disentangling monocular 3d object detection. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Korea, 27 October–2 November 2019; pp. 1991–1999.
50. Rezatofighi, H.; Tsoi, N.; Gwak, J.; Sadeghian, A.; Reid, I.; Savarese, S. Generalized intersection over union: A metric and a loss for bounding box regression. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seoul, Korea, 27 October–2 November 2019; pp. 658–666.
51. Këpuska, V.; Bohouta, G. Comparing speech recognition systems (Microsoft API, Google API and CMU Sphinx). *Int. J. Eng. Res. Appl.* **2017**, *7*, 20–24. [CrossRef]
52. Dilmegani, C. Best OCR by Text Extraction Accuracy in 2022. Available online: https://research.aimultiple.com/ocr-accuracy/ (accessed on 31 May 2022).
53. Rivest, R. *The MD5 Message-Digest Algorithm*; Technical report; MIT Laboratory for Computer Science: Cambridge, MA, USA, 1992.
54. Brownlee, J. A gentle introduction to transfer learning for deep learning. *Mach. Learn. Mastery* **2017**, *20*. Available online: https://machinelearningmastery.com/transfer-learning-for-deep-learning/ (accessed on 31 May 2022).