International Journal of Scientific Research and Engineering Development-– Volume 7 Issue 2, Mar- Apr 2024
Available at www.ijsred.com

RESEARCH ARTICLE OPEN ACCESS

A Model to Determine Hashtags for Social Media Video Clips Using Computer Vision Techniques

Jayanga Palihena[1], W.A. Lahiru Randika[2]
[1](School of Artificial Intelligence, Nanjing University of Information Science and Technology, Nanjing, China
Email: jayanga.sl@gmail.com)
[2](School of Artificial Intelligence, Nanjing University of Information Science and Technology, Nanjing, China
Email: lahirurandika@hotmail.com)

--*************************----------------------------------

Abstract*:*
In the present day online social media content is fast growing area, due to this the organization and retrieval of content has become increasingly challenging to its users; thus, hashtags are used. Hashtags are a powerful tool used for organizing and categorizing social media content, where users can search and navigate for content based on a given hashtag. The study presents the development and evaluation of a novel model architecture that combines both CNN and RNN (LSTM) to classify videos and predict the hashtags accordingly, which uses a modified version of the UFT101 dataset, where the normal labels have been replaced with hashtags. The hashtags for the label have been obtained from the web, by using the web scraping methodology to get hashtags for a given keyword. The model is prepared to learn complex relationships between visual substances and compare hashtags. Moreover, hyperparameter tuning, dropout layers, and regularization techniques are incorporated to enhance the model's accuracy and generalization capabilities.

***Keywords* — Video classification, RNN (Recurrent Neural Networks), CNN (Convolutional Neural Networks), LSTM, Hashtags, Dropout layers, Feature extraction, UFT101, Video clips**

--*************************----------------------------------

I. INTRODUCTION

Social media is still a vital component of contemporary communication in 2024, fostering hitherto unheard-of connections between people, groups, and companies. The social media ecosystem has evolved dramatically in response to the swift advancement of technology and shifting societal norms, profoundly influencing our interactions, information sharing, and worldview. The number of social media users worldwide has surpassed the 4.5 billion marks, demonstrating the ubiquitous influence of these platforms around the world [1]. The users are presented with a wide range of options to interact with content and build connections, from immersive VR experiences to fleeting content on social media sites like Facebook, WeChat, TikTok, and Snapchat. As of 2024 January, there are around 5.3 billion Internet users worldwide which covers over 66.2 percent of the global population [2]. The usage of social media extends far beyond mere communication, where it is impacting various other sectors such as education, fashion, transportation, entertainment, healthcare, agriculture, and numerous others. For instance, in the field of education, platforms like Twitter and Facebook are increasingly utilized by educators to foster collaboration, share resources, and engage students in innovative learning experiences[3].

In addition, social media has transformed healthcare, enabling patients to access medical information, engage with healthcare professionals, and participate in online support communities[4]. In agriculture, platforms such as YouTube and WhatsApp have provided farmers with valuable insights, market information, and peer support networks, improving the productivity and sustainability of farming practices[5]. As a result, social media's multifarious effects highlight how essential it is as a spark for creativity and connection among many businesses.

Hashtags have become indispensable tools for interaction, discovery, and classification in the ever-expanding world of online communication and social media. Initially, hashtags were first used in the Twitter platform, but now hashtags are popular on various other social media platforms like Instagram, Facebook, LinkedIn, TikTok, WeChat, etc. A hashtag is a word or expression went before by the pound (#) image, which effectively sorts out content and makes it easily searchable for users, for example, #StopSmoking, #HIV, or #WeStayTogether are used in social media posts[6].

Hashtags allow users to categorize their social media content or posts around a theme, that post could be an image, video, blog, tweet, or any message using a hashtag will connect that content with a word or phrase.

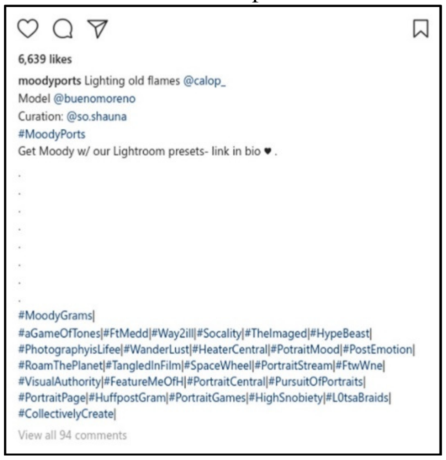

6,639 likes
moodyports Lighting old flames @calop_
Model @buenomoreno
Curation: @so.shauna
#MoodyPorts
Get Moody w/ our Lightroom presets- link in bio ♥ .
.
.
.
.
.
.
.
.
#MoodyGrams|
#aGameOfTones|#FtMedd|#Way2ill|#Socality|#TheImaged|#HypeBeast|
#PhotographyisLifee|#WanderLust|#HeaterCentral|#PotraitMood|#PostEmotion|
#RoamThePlanet|#TangledInFilm|#SpaceWheel|#PortraitStream|#FtwWne|
#VisualAuthority|#FeatureMeOfH|#PortraitCentral|#PursuitOfPortraits|
#PortraitPage|#HuffpostGram|#PortraitGames|#HighSnobiety|#L0tsaBraids|
#CollectivelyCreate|
View all 94 comments

Fig. 1 Usage of Hashtags in Social Media Platforms

This study aims to develop a model to automate the process of generating hashtags for video clips shared on social media platforms. This will be accomplished using computer vision technologies to predict hashtags based on the given short video. These video clips, typically no longer than 15 seconds, are a common format on social media. The proposed model capitalizes on this format's brevity to provide succinct and meaningful hashtags. Users will find this model to be an invaluable tool as they share their videos online. Upon uploading a video, the model springs into action. It analyses the video's content and predicts hashtags that are closely aligned with the video's themes. This seamless integration of technology assists users in making their content more discoverable.

II. Related Work

Numerous publications and research works have conducted for video classification, using a variety of methodologies. These investigations span a wide array of use cases, including object detection, action detection, sports analysis, crime detection, among others. In this chapter, my objective is to review previous research works conducted prior to my study, delving into their methodologies and outcomes.

A. *Video Classification*

Video classification is a part of computer vision that automatically categorizes videos based on their content and sorts the videos into their respective categories. For example, actions can include various activities like dancing, walking, or running, while behavioural emotions may encompass feelings such as cheerfulness, sadness, or surprise. Image classification is done based on the spatial content, for example, a picture of a human being vs a picture of an animal, whereas video classification is done based on the spatial and temporal features. A normal video consists of many frames, each containing a large amount of information. The video also contains different lighting conditions, from different angles and different frame rates as well[7].

During video classification there few changes that need to be overcome:

- Scalability: When the volume of data increases, it becomes challenging to process and classify them in a reasonable time frame.
- Video annotation: Labelling large amounts of video data is a very time-consuming and tedious task to be done.
- Generalization: The majority of the video classification algorithms are implemented to work on a specific dataset and are not generalized to work on other datasets.
- Privacy and security: There are many privacy and security concerns when it comes to video data such as facial recognition and surveillance.
- Video quality: Video clips come in diverse sizes and qualities, presenting challenges in handling them that may compromise the performance of the video classifier.

B. *Convolutional Neural Networks (CNNs)*

Drawing inspiration from the visual perception mechanisms of animals and the McCulloch-Pitts model Fukushima introduced the “neocognitron” in 1980, which was the first computational model to use connectivity between neurons of a hierarchically transformed image (Fukushima 1980). Experiments were conducted by applying neurons with similar parameters to the patch from the previous layer at different locations, simultaneously this approach is considered as the precursor to convolutional neural networks (CNN). The modern framework of CNN and LeNet-5 showcased modern-day performance. LeNet-5 uses multiple layers and is trained using the back-propagation algorithm in an end-to-end manner. This involves directly classifying visual patterns using raw images. However, due to constraints such as the scarcity of labelled training data and computational limitations, LeNet-5 and its variations struggled to achieve satisfactory performance on more intricate vision tasks until recent advancements.

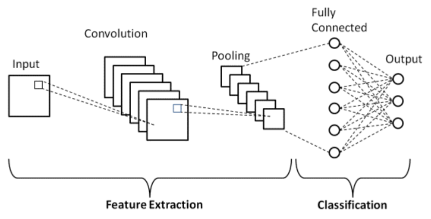


Fig. 2 Basic CNN Architecture

Fig. 2 illustrates a convolutional mechanism used to dissect and categorize the distinct attributes of an image during the process known as Feature Extraction. This process involves a network comprised of numerous convolutional or pooling layer pairs. Subsequently, a fully connected layer utilizes the outcomes of the convolutional process to predict the image's class based on the features extracted in earlier stages. The objective of this CNN-based feature extraction model is to diminish the number of features within a dataset. It generates novel features that encapsulate the essence of the original set of features.

C. *Recurrent Neural Networks (RNNs)*

To better inspect the temporal and sequential features recurrent connection structures have been added, forwarding to the emergence of recurrent neural networks (RNNs). RNN’s allow regular connections to form cycles, where enabling a memory of previous inputs to persist in the network's internal state [8].

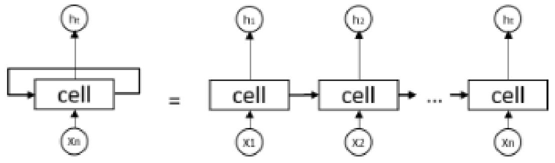


Fig. 3 Basic RNN Structure

The architecture of an RNN consists of recurrent connections that allow information to grow over time, which allows the network to keep the memory of the past inputs while processing the current ones. The recurrent nature enables RNN's to handle input sequences of differing lengths and extract meaningful patterns and features from them. This arcane prowess finds application in endeavours as diverse as the seamless deciphering of unsegmented, interconnected handwriting, where it claims the mantle of unrivalled achievement.

D. *Long-term memory (LSTM)*

It is implemented in LSTM networks in addition to using a portion of the prior output for new processing. When information is processed through all the inputs, it moves from cell to cell in long-term memory with little change in certain features. The long-term coherence of the forecasts is made possible by this continuous input[9].

$$\boldsymbol{i}^{(t)} = \sigma\left(W_{xi}\boldsymbol{x}^{(t)} + W_{hi}h^{(t-1)} + W_{ci}\boldsymbol{c}^{(t)} + \boldsymbol{b_i}\right),$$
$$\boldsymbol{f}^{(t)} = \sigma\left(W_{xf}\boldsymbol{x}^{(t)} + W_{hf}h^{(t)} + W_{ci}\boldsymbol{c}^{(t)} + \boldsymbol{b_f}\right),$$
$$\boldsymbol{c}^{(t)} = f^{t}c^{(t-1)} + i, \tanh(W_{xc}x^{(t)} + W_{hc}x^{(t-1)}) + \boldsymbol{b_c}),$$
$$\boldsymbol{o}^{(t)} = \sigma\left(W_{xo}\boldsymbol{x}^{(t)} + W_{ho}h^{(t-1)} + W_{co}\boldsymbol{c}^{(t)} + \boldsymbol{b_o}\right),$$
$$\boldsymbol{h}^{(t)} = o^{t}\tanh\left(c^{(t)}\right)\#(1)$$

In the LSTM model, denoted by $x^{(t)}$ and $h^{(t)}$ for input and hidden vectors at time step t, respectively, activation vectors $i^{(t)}$, $f^{(t)}$, $c^{(t)}$, and $o^{(t)}$ represent the input gate, forget gate, memory cell, and output gate. The weight matrix between two vectors is represented by Wαβ, where βand α denote the vectors. For instance, Wαβ denotes the weight matrix from input $x^{(t)}$ to input gate $i^{(t)}$.

Long-term memory is implemented in LSTM networks in addition to using a portion of the prior output for new processing. When information is processed through all the inputs, it moves from cell to cell in long-term memory with little change in certain features. The long-term coherence of the forecasts is made possible by this continuous input[9].

E. *Sequence Learning*

Sequence learning is a fundamental concept in the field of artificial intelligence and machine learning, particularly in tasks involving data with a sequential nature, such as time series data, natural language processing, and video analysis. The objective of sequence learning algorithms is to understand and extract meaningful patterns or dependencies present in sequential data.

One of the key components of sequence learning is Recurrent Neural Networks (RNNs), which are designed to process sequences of data by maintaining an internal state or memory. RNNs are particularly effective in capturing temporal dependencies within sequential data, making them well-suited for tasks such as language modelling, speech recognition, and video analysis.

III. METHODOLOGY

The methodology used in this study focuses on making use the power of video classification techniques to automate the process of hashtag prediction for video clips. In the contemporary digital landscape, where video content proliferates across various online platforms, the effective categorization and tagging of videos play a major role in enhancing discoverability and engagement. By leveraging advanced machine learning algorithms, particularly video classification models, this research aims to streamline the hashtag generation process, facilitating efficient content organization and retrieval in multimedia environments.

F. *Problem Identification*

In today's digital era characterized by the exponential growth of video content across online platforms, the process of organizing and categorizing vast amounts of video data presents significant challenges. Traditional methods of manual tagging and labelling are time-consuming, labour-intensive, and prone to inconsistencies. The process of manually assigning effective and trending hashtags to video clips on social media platforms remains laborious and inherently subjective[10]. While advancements have been made in automatic hashtag prediction for various content formats, such as text and images[11]. There is a critical gap in research specifically focusing on utilizing video classification for generating hashtags tailored to video clips.

Considering these challenges, there is a critical need for a solution that harness the power of machine learning and video classification techniques to automate the generation of descriptive metadata, particularly hashtags, for video clips. The proposed methodology aims to bridge this gap by investigating the potential of leveraging video classification techniques to automatically generate relevant and trending hashtags specifically for video clips. By using video classification methodology, this research seeks to empower content creators with efficient hashtag generation tools, ultimately contributing to increased discoverability and engagement for their video content.

G. *Dataset Preparation*

The foundation of this experiment rests on the UFT 101 dataset, The UFT 101 dataset serves as the cornerstone of numerous research endeavours in the field of video classification and analysis. Originally introduced for action recognition tasks, this dataset has been widely adopted and extended to explore various aspects of video understanding.

Which has been tailored for this study by integrating hashtags. Since there was no dataset set readily available for with action/scene detection that related with hashtags the dataset had to be modified, traditional labels have been substituted with hashtags that aptly capture the essence of each scene, thereby enriching the dataset with contextual information.

The hashtags for the relevant video label were obtained from the existing web sites and other internet resources. The process of hashtag scraping from the web involves extracting hashtags associated with specific keywords or topics from various online sources such as social media platforms, forums, or websites. This can be achieved using web scraping techniques, where automated scripts or tools are used to gather relevant information from web pages.

Web scraping is a technique used to extract data from websites automatically. In the context of this study, web scraping is employed to collect hashtags from various online platforms. The process involves accessing web pages, parsing the HTML content, and extracting relevant information, such as hashtags, for further analysis.

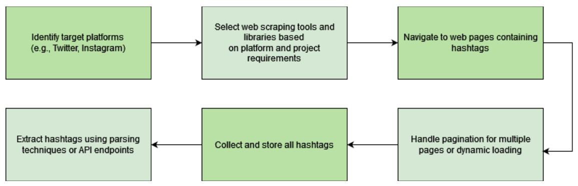

Identify target platforms (e.g., Twitter, Instagram)
Select web scraping tools and libraries based on platform and project requirements
Navigate to web pages containing hashtags
Handle pagination for multiple pages or dynamic loading
Collect and store all hashtags
Extract hashtags using parsing techniques or API endpoints

Fig. 4 Flow of Web Scraping for Hashtags

The scraping process begins with identifying the target platforms where hashtags are to be collected. Popular social media platforms like Twitter, Instagram, and Facebook are common sources of hashtags. Once the platforms are identified, appropriate web scraping tools and libraries are selected based on the nature of the platforms and project requirements.

H. Video Loading and Processing

Before the feature extraction the video clips from the dataset need to be resized and cropped, in order to perform this task OpenCV library has been used.

First, a frame (image) is taken as input and crops it to form a square with its centre as the focal point. It calculates the dimensions of the input frame and determines the minimum dimension (either width or height). Then, it computes the starting coordinates for cropping based on the centre of the frame and the minimum dimension. Finally, it returns the cropped square frame.

The video files are being loaded. It initializes a video capture object using OpenCV's VideoCapture class. It then iterates through the video frames using a while loop, reading each frame using cap.read(). Inside the loop, it checks if the frame was successfully read (ret is True), and if so, it crops the frame to form a square using the crop_center_square function. After cropping, it resizes the frame to a specified size using OpenCV's resize function. Additionally, it reorders the colour channels of the frame from OpenCV's default order (BGR) to RGB. The processed frame is then appended to a list of frames. The max_frames are set to non-zero value and the number of frames loaded reaches this limit.

I. Feature Extraction

Feature extraction is a major role in computer vision tasks by transforming raw input data, such as images or videos, into a more compact and meaningful representation. These extracted features capture essential patterns, structures, or characteristics of the input data, enabling subsequent analysis, recognition, or classification tasks. Feature extraction algorithms aim to identify and represent relevant information present in the visual data efficiently. Raw data, in its unprocessed form, often contains irrelevant or redundant information that can hinder the performance of machine learning algorithms.

This study proposes a convolutional neural network (CNN) architecture designed for feature extraction from video clips. The breakdown of the architecture is as follows:

- Input Layers: The input tensor, which represents the image data. It typically takes the form (height, width, channels).Height and width represent the dimensions of the input images.Channels denotes the number of color channels in the images (e.g., 3 for RGB images, 1 for grayscale).
- Convolutional Layers: The model starts with a series of convolutional layers, each followed by a rectified linear unit (ReLU) activation function. Convolutional layers apply learnable filters to the input image, enabling the extraction of spatial features such as edges, textures, and patterns.

$$(\mathrm{I} * \mathrm{K})(\mathrm{i},\mathrm{j}) = \sum_{m}\sum_{n} I(m,n)\cdot K(i-m,j-n) \quad \#(2)$$

Where:
I represent the input image
K denotes the filter/kernel.
(I∗K)(i,j) denotes the result of the convolution operation at position (i,j)(i,j)
(m,n) iterates over the spatial dimensions of the input image.
(i,j): denotes the spatial position in the output feature map.

- Max Pooling: Partitions the input feature map into a set of non-overlapping rectangular pooling regions and, for each region, outputs the maximum value.

$$MaxPooling(x,y) = max_{i\in pooling\ region}(x_i,y_i) \quad \#(3)$$

Where:
x,y denote the spatial coordinates of the pooling region.
(x_i,y_i) represents the values within the pooling region.
MaxPooling(x,y) denotes the output of the MaxPooling operation at position (x,y)(x,y).

- Global Average Pooling Layer: This layer computes the average value of each feature map across all spatial locations, resulting in a fixed-length vector regardless of the input image size.

- Activation Function: After convolution, the output is passed through an activation function, typically the Rectified Linear Unit (ReLU) activation function, which introduces non-linearity into the model by mapping negative values to zero and leaving positive values unchanged.

$$f(x) = max(0, x) \#(4)$$

- Output Feature Maps: The output feature maps produced by the convolutional layers represent the presence of specific patterns or features within the input images. These feature maps are then passed.
- Model Compilation: Once the layers are defined, the model is compiled with appropriate loss function, optimizer, and metrics for training.to subsequent layers for further processing, such as pooling and classification.

J. Label Processor

The label processor is a crucial component in the preprocessing pipeline of deep learning models, particularly in tasks involving categorical data, such as classification. This layer plays a major role in converting categorical labels, often represented as strings, into a numerical format suitable for model training. By encoding labels into integer indices, the label processor ensures consistency in label representation across training samples and facilitates efficient handling of categorical data during model training. This introductory layer serves as a bridge between the raw categorical labels and the numerical computations performed by deep learning models, laying the groundwork for effective model training and predictive accuracy.

The label processor seamlessly integrates with deep learning architectures, providing a seamless transition from string-based labels to numerical inputs for the model. During training, the model receives integer-encoded labels as targets, enabling it to learn the mapping between input data and corresponding labels effectively. Furthermore, it serves as a foundational component in the preprocessing pipeline of deep learning models, ensuring the compatibility, efficiency, and effectiveness of categorical label data in various classification and prediction tasks.

In this study the label processor layer is configured to map hashtags (strings) from the training data to integer indices, facilitating the encoding of categorical labels for training deep learning models. It ensures consistency in label representation across training samples and enables efficient handling of categorical data during model training.

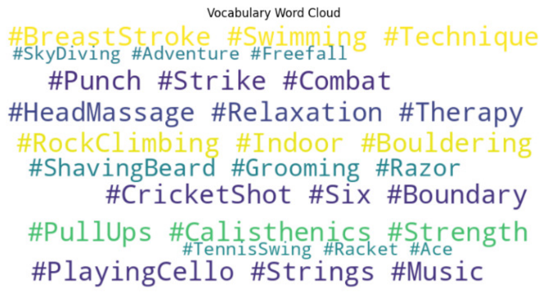


Fig. 5 Word Vocabulary Based on the Dataset

This is word vocabulary used in this experiment based on the dataset labels. The word vocabulary is utilized in the experiment to initialize the label processor, which is responsible for mapping string-based labels to numerical indices. The word vocabulary influences how labels are encoded into numerical representations during model training. During training, the model receives these encoded labels as targets, enabling it to predict the corresponding label indices for input data.

K. Sequence Preparation

This is a critical step in the pipeline of processing video data for training sequence models, such as recurrent neural networks (RNNs) or transformers. Video data, inherently sequential in nature, poses unique challenges due to variable-length sequences of frames in each video. The process involves organizing the extracted features from video frames into structured sequences of fixed length, ensuring compatibility with sequence model architectures. Additionally, sequence preparation involves the application of masking techniques to handle sequences of varying lengths, facilitating uniform input dimensions during training. This crucial preprocessing step lays the foundation for effective training and analysis of video data using sequence models, enabling tasks such as video classification, action recognition, and video captioning.

In this study, we discuss the methodology and techniques involved in sequence preparation, illustrating its significance in the realm of video analysis and deep learning. Fig. 6shows the architecture of the CNN's model:

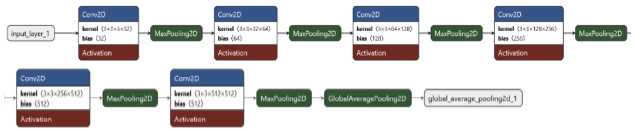


Fig. 6 Custom CNN Model Architecture

- Input Layers: Starting point of the model receiving input image data. Each pixel of the image is treated as a feature. Dimensions correspond to the image size (e.g., height, width, and color channels).
- Conv2D Layers:

- Perform convolution operations using filters (kernels) to extract local patterns and create feature maps.
- First Conv2D layer: 3x3 kernel, producing 32 feature maps with 32 biases.
- Second Conv2D layer: 3x3 kernel, outputting 64 feature maps with 64 biases.
- Third Conv2D layer: Continues process, resulting in 128 feature maps with 128 biases.
- Fourth Conv2D layer: Generates 256 feature maps with 256 biases.
- Fifth Conv2D layer: Applies filter to output 512 feature maps.

- Activation Layers: Introduce non-linearity using activation functions like ReLU (Rectified Linear Unit). ReLU sets negative values to zero, enabling learning of complex patterns.
- MaxPooling2D Layers: Reduce spatial dimensions (width and height) of feature maps. Max-pooling selects maximum value within small window (e.g., 2x2) and down-samples feature maps. Makes detected features more robust to scale and orientation changes
- GlobalAveragePooling2D Layer: Further reduces each feature map to a single number by taking average of all values. Prepares features for final classification layer

L. LSTM Model Sequence Learning

This chapter represents a fundamental approach to address such tasks, leveraging the inherent capability of LSTM networks to capture long-term dependencies in sequential data. This introduction aims to provide an overview of the LSTM model's role in sequence learning, highlighting its architecture, capabilities, and applications across various domains.

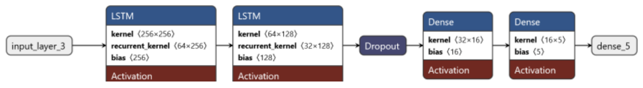


Fig. 7Custom LSTM Architecture

Fig. 7 shows the flow of the LSTM model and layers which are used with the model training. Creating a custom LSTM layer for sequence learning, within the TensorFlow/Keras framework, essential for sequence learning tasks in deep learning. The CustomLSTM class encapsulates this custom layer definition, offering a flexible and configurable solution tailored to specific model requirements. Upon initialization, parameters such as the number of units, dropout rates, and regularization techniques are specified, enabling fine-tuning of the layer's behavior. During the building phase, the layer's weights, including kernel weights, recurrent kernel weights, and biases, are initialized based on the input shape. The forward pass of the layer, defined within the call method, iteratively processes input sequences, computes output sequences using LSTM equations, and applies dropout regularization to input and recurrent units if specified. Additionally, the compute_output_shape method ensures proper output shape calculation based on input dimensions and layer configurations. Overall, this custom LSTM layer serves as a versatile building block for constructing deep learning models tailored to sequence learning tasks, facilitating the exploration and advancement of various applications, including natural language processing, time-series prediction, and video data analysis.

The get_sequence_model function encapsulates the construction of a Sequential model tailored for sequence classification tasks. Beginning with the definition of a L2 regularization instance, the function establishes a strategy for penalizing large weights, thereby mitigating overfitting. Subsequently, the model architecture is built layer by layer within a Sequential container. The first layer instantiated is a custom LSTM layer configured with 64 units, set to return sequences to preserve temporal information, and adorned with L2 regularization on its kernel weights. Following this, another custom LSTM layer with 32 units is added, devoid of sequence return, yet similarly imbued with L2 regularization. Introducing a Dropout layer with a dropout rate of 0.5, the model incorporates a regularization mechanism to stave off overfitting by randomly deactivating units during training. Subsequent to the LSTM layers, two Dense layers are appended to the model, each adorned with ReLU activation and L2 regularization, facilitating non-linear transformations and contributing to the model's robustness. Finally, the function returns the assembled model, equipped to undertake sequence classification tasks with efficacy and generalization prowess.

The class initializes the weights required for the LSTM layer based on the input shape. Firstly, the dimensionality of the input data, denoted as self.input_dim, is extracted from the last dimension of the input shape. This dimension represents the feature dimensionality of the input sequences. Subsequently, the kernel weights (self.kernel) are instantiated as trainable parameters, with a shape determined by the product of the input dimension and four times the number of units in the layer. The kernel weights are initialized using the He normal initializer, aiming to preserve the variance of the activations through the layers. Additionally, a regularization term may be applied to the kernel weights to control overfitting, denoted by self.kernel_regularizer. The recurrent kernel weights (self.recurrent_kernel) are created with a shape determined by the number of units in the layer, initialized using the orthogonal initializer to facilitate gradient flow during training. Similar to the kernel weights, a regularization term may be incorporated for the recurrent kernel weights (self.recurrent_regularizer). Finally, biases (self.bias) are introduced as trainable parameters, initialized with zeros to introduce a shift in the activation function. These weights and biases are essential components of the LSTM layer, governing the transformation and propagation of information through the recurrent network. Upon completion of weight initialization, the build method concludes by calling the superclass's build method to finalize the layer construction process.

IV. EXPERIMENTS AND RESULTS

The experiments were conducted using Pytorch, TensorFlow and Google Collab platform as the primary environment. The Google Collab was utilized for computing power with Nvidia GPUs for training and testing purposes.

The initial testing was done using the UFT101 dataset with the modified labels with hashtags. The below table shows 10 random rows from the dataset before the data preprocessing stage.

TABLE I
UFT101 CLIP NAMES AND HASHTAG LABELS

Clip_name	Hashtag
v_ShavingBeard_g16_c02.avi	#ShavingBeard #Grooming #Razor
v_TennisSwing_g11_c01.avi	#TennisSwing #Racket #Ace
v_TennisSwing_g15_c03.avi	#TennisSwing #Racket #Ace
v_PlayingCello_g24_c05.avi	#PlayingCello #Strings #Music
v_Punch_g03_c03.avi	#Punch #Strike #Combat
v_Punch_g19_c07.avi	#Punch #Strike #Combat
v_PlayingCello_g25_c06.avi	#PlayingCello #Strings #Music
v_CricketShot_g16_c07.avi	#CricketShot #Six #Boundary
v_ShavingBeard_g17_c03.avi	#ShavingBeard #Grooming #Razor
v_CricketShot_g21_c01.avi	#CricketShot #Six #Boundary

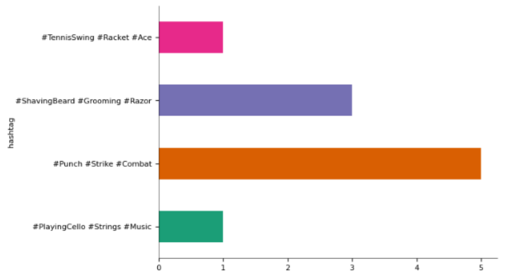


Fig. 8 UFT101 Dataset Labels

M. Dataset Splitting

Training Dataset (714 video clips): This dataset consists of 714 video clips that are used to train the machine learning model. Each video clip likely contains a sequence of frames representing certain actions such as shaving beard, basketball playing, cricket playing and etc. During training, the model learns from these video clips by adjusting its parameters to minimize a loss function, typically based on a comparison between its predictions and the ground truth labels associated with each video clip.

Testing Dataset (344 video clips): This dataset, separate from the training data, comprises 344 video clips that are used to evaluate the trained model's performance. The model hasn't seen these video clips during training.

N. Experiments and Results

The training process was configured to run for a total of 30 epochs (epochs=30), allowing the model to undergo multiple iterations over the training dataset, progressively refining its predictive capabilities. Additionally, to manage computational resources effectively and expedite the training process, I employed a batch size of 15 (batch_size=15). This parameter determined the number of samples processed by the model before updating its weights, facilitating efficient gradient descent optimization.

Furthermore, to enhance training efficiency and prevent overfitting, I incorporated an early stopping mechanism. This was achieved through the inclusion of a callback function (early_stopping), which monitored the validation loss and halted training if no improvement was observed over a predefined number of epochs (patience=3). By implementing this mechanism, I ensured that the model's performance was optimized while mitigating the risk of overfitting to the training data.

The proposed model achieved a validation accuracy of 71.43% during the training and validation process.

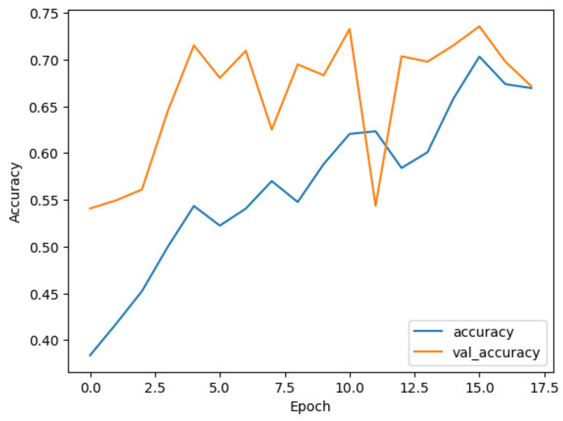


Fig. 9 Validation and Test Accuracy of the Hashtag Model

TABLE II
ACCURACY AND LOSS OF EPOCHS AND VALIDATION SETS

Epoch	Accuracy	Loss	Val_Accuracy	Val_Loss
1	0.3684	4.4627	0.5407	2.9259
2	0.4337	2.62	0.5494	2.2309
3	0.441	2.1074	0.561	1.8773
4	0.4883	1.7638	0.6453	1.6674
5	0.5711	1.5545	0.7151	1.4914
6	0.4725	1.6274	0.6802	1.4332
7	0.5055	1.558	0.7093	1.3501

8	0.5813	1.3603	0.625	1.2879
9	0.5092	1.4728	0.6948	1.3005
10	0.5736	1.2638	0.6831	1.206
11	0.6134	1.2604	0.7326	1.1656
12	0.6501	1.1793	0.5436	1.278
13	0.4959	1.3733	0.7035	1.2182
14	0.6026	1.2051	0.6977	1.0913
15	0.6541	1.1668	0.7151	1.0581
16	0.6819	1.0713	0.7355	1.0928
17	0.6645	1.1102	0.6977	1.1487
18	0.7143	1.0018	0.6715	1.1437

Table 2 shows the break drown of how the model performed in epoch.

- The accuracy starts at around 36.84%, the loss is relatively high at 4.4627, but the validation accuracy is higher at 54.07% with a lower validation loss of 2.9259.
- The accuracy improves slightly to 43.37% with a decrease in loss to 2.6200. Validation accuracy increases to 54.94%, and validation loss decreases further to 2.2309.
- The accuracy continues to improve to 44.10%, with the loss decreasing to 2.1074. Validation accuracy increases to 56.10%, and validation loss decreases to 1.8773.

This pattern continues across the epochs, with the model's accuracy and loss improving gradually. In the end, the model achieves an accuracy of approximately 71.43% with a loss of 1.0018, while the validation accuracy is around 67.15% with a validation loss of 1.1437.

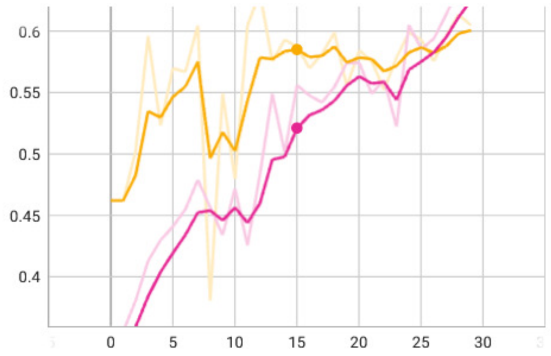


Fig. 10 Epoch Accuracyddp8022

Fig. 2 shows the epoch accuracy during the training process.On the x-axis, we observe the number of epochs, ranging from 0 to 30. The y-axis represents accuracy, which varies between approximately 0.4 and 0.6. Two lines are plotted on the graph: a yellow line characterized by fluctuations and a pink line that exhibits smoother progress. Both lines start with modest accuracy but steadily improve as training proceeds.

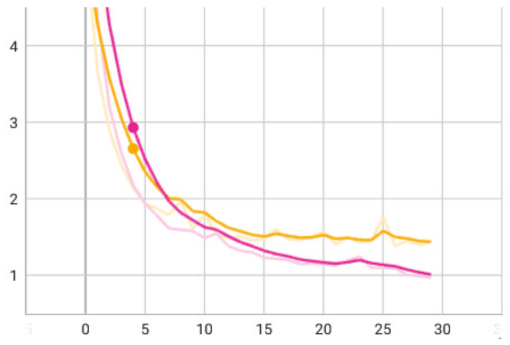


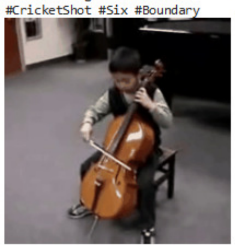
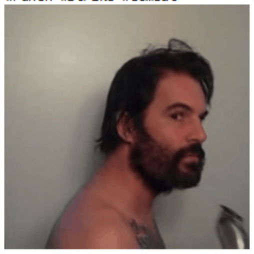
Fig. 11 Epoch Loss

Fig. 3 shows the epoch loss during the training process.On the x-axis, we observe the number of epochs, ranging from 0 to 30. The y-axis represents the loss (also known as the error), which varies between approximately 0.4 and 0.6. Three lines are plotted on the graph, each in a different colour. Initially, all lines start with higher loss values at epoch 0 and decrease sharply until around epoch 5. A pinkish-purple dot highlights an intersection point on one of the lines around epoch number five and a loss value of two. After epoch five, all lines show a more gradual decline or plateau in loss values.

O. Test Results

Table presents the final outcomes of the hashtag prediction model, wherein it forecasts hashtags corresponding to the provided video clips. This indicates the output or outcome of the model after it has been trained and tested. It represents the model's performance in predicting hashtags for the given video clips. The model offers three hashtags for each video, each accompanied by four associated probabilities. The model predicts three hashtag predictions for each video clip. This suggests that the model doesn't just output a single hashtag but provides a set of potential hashtags that could be relevant to the content of the video. Along with each predicted hashtag, the model assigns four probabilities. These probabilities indicate the likelihood or confidence of each hashtag being relevant to the video clip. Having four probabilities might suggest that the model incorporates some form of uncertainty estimation or confidence scoring for its predictions.

TABLE III
VIDEO CLIP CLASSIFICATION RESULTS BY HASHTAGS

Test	Result
Video clip of two people fighting	Hashtags Probability ------------------------------ ------------ #Punch #Strike #Combat 95.52% #PlayingCello #Strings #Music 3.65% #ShavingBeard #Grooming #Razor 0.53% #TennisSwing #Racket #Ace 0.24% #CricketShot #Six #Boundary 0.06%
Video clip of a cricket match	Hashtags Probability ------------------------------ ------------ #CricketShot #Six #Boundary 66.75% #TennisSwing #Racket #Ace 24.15% #ShavingBeard #Grooming #Razor 7.14% #PlayingCello #Strings #Music 1.18% #Punch #Strike #Combat 0.78%
Video clip of a kid playing a cello	Hashtags Probability ------------------------------ ------------ #PlayingCello #Strings #Music 39.20% #ShavingBeard #Grooming #Razor 33.38% #Punch #Strike #Combat 12.75% #TennisSwing #Racket #Ace 10.94% #CricketShot #Six #Boundary 3.74% 
Video clip of a person shaving his beard	Hashtags Probability ------------------------------ ------------ #ShavingBeard #Grooming #Razor 40.70% #CricketShot #Six #Boundary 22.16% #TennisSwing #Racket #Ace 18.22% #PlayingCello #Strings #Music 14.23% #Punch #Strike #Combat 4.70% 

P. Comparison with Existing Method

The hashtag prediction model's performance was assessed through a comparative analysis against existing research on video classification methods, with accuracy as the primary metric for evaluation. Table 4 compares the proposed hashtag prediction model with existing research models related to video classification.

TABLE IV
COMPARISON WITH EXISTING VIDEO CLASSIFICATION APPROACHES

Model	Accuracy
Hashtag Prediction Model – UFT101	71.51%
FASTER32 - HMDB-51 [12]	75.70%
T3D-Transfer - HMDB51[13]	61.10%
T3D+TSN - HMDB5 [13]	63.50%
DOVF- HMDB51 [14]	71.70%
Objects2action - UCF101 [15]	63.90%

The above test result for a different model compared with our hashtag prediction model, The hashtag prediction model achieved an accuracy of 71.51%, indicating that it correctly predicted approximately 71.51% of the hashtags for the given set of video clips. Comparative analysis reveals that the hashtag prediction model outperforms some existing methods (T3D-Transfer, T3D+TSN, Objects2action), achieving higher accuracy scores than these models. As illustrated the model has better accuracy compared to other research work which has been conducted with different datasets and various other approaches.

V. CONCLUSIONS

This study has explored the task of predicting hashtags for video clips using video classification techniques, focusing on the utilization of Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs).

Through experimentation conducted on the UTF101 dataset, we employed a combination of feature extraction and sequence learning methodologies. This approach allowed us to capture both spatial and temporal information inherent in video data, enabling a more comprehensive understanding of the content. The results demonstrate the potential of CNNs and RNNs in effectively discerning the semantic context of video clips and generating meaningful hashtags accordingly.

In conclusion, while this study makes significant strides in leveraging video classification techniques for automated hashtag generation, several challenges and opportunities for future research remain. By addressing these challenges and fostering interdisciplinary collaboration, researchers can continue to push the boundaries of innovation in video understanding and content annotation, ultimately advancing the field towards more intelligent and context-aware multimedia systems.

ACKNOWLEDGMENT

We would like to thank all who helped us with their time and dedication. Their insightful feedback and guidance greatly improved the quality of my work. We also extend my heartfelt gratitude to my family, friends and batchmates for their unwavering support and encouragement throughout this journey.We would also like to express our gratitude to Mr.LalinduWeerakkody for support given to us through out this research.

Thank you all for your contributions to this research.

REFERENCES

[1] Statista, “Number of social media users worldwide from 2017 to 2025.,” https://www.statista.com/.

[2] “Internet and social media users in the world 2024 | Statista.” Accessed: Feb. 27, 2024. [Online]. Available: https://www.statista.com/statistics/617136/digital-population-worldwide/

[3] E. E. Smith, “Social media in undergraduate learning: categories and characteristics,” *International Journal of Educational Technology in Higher Education*, vol. 14, no. 1, Dec. 2017, doi: 10.1186/S41239-017-0049-Y.

[4] S. A. Moorhead, D. E. Hazlett, L. Harrison, J. K. Carroll, A. Irwin, and C. Hoving, “A New Dimension of Health Care: Systematic Review of the Uses, Benefits, and Limitations of Social Media for Health Communication,” *J Med Internet Res 2013;15(4):e85 https://www.jmir.org/2013/4/e85*, vol. 15, no. 4, p. e1933, Apr. 2013, doi: 10.2196/JMIR.1933.

[5] A. Kamilaris, A. Fonts, and F. X. Prenafeta-Boldύ, “The rise of blockchain technology in agriculture and food supply chains,” *Trends Food Sci Technol*, vol. 91, pp. 640–652, Sep. 2019, doi: 10.1016/J.TIFS.2019.07.034.

[6] R. Goyena, “#AdvocatingForChange: The Strategic Use of Hashtags in Social Media Advocacy 1,” *J Chem Inf Model*, vol. 53, no. 9, pp. 1689–1699, 2019, doi: 10.1017/CBO9781107415324.004.

[7] “Video Classification: Methods, Use Cases, Tutorial.” Accessed: Mar. 03, 2024. [Online]. Available: https://www.v7labs.com/blog/video-classification-guide

[8] A. Graves, “Supervised Sequence Labelling with Recurrent Neural Networks,” vol. 385, 2012, doi: 10.1007/978-3-642-24797-2.

[9] M. Camargo, “Learning Accurate LSTM Models of Business Processes,” 2019.

[10] B. Knowles and J. T. Richards, “ACM TechBrief: Trusted AI,” *ACM TechBrief: Trusted AI*, Jan. 2024, doi: 10.1145/3641524.

[11] W. Zaremba, I. Sutskever, O. Vinyals, and G. Brain, “Recurrent Neural Network Regularization,” Sep. 2014, Accessed: Mar. 07, 2024. [Online]. Available: https://arxiv.org/abs/1409.2329v5

[12] L. Zhu *et al.*, “FASTER Recurrent Networks for Efficient Video Classification,” 2020. [Online]. Available: www.aaai.org

[13] A. Diba*et al.*, “Temporal 3D ConvNets: New Architecture and Transfer Learning for Video Classification,” 2018, Accessed: Mar. 13, 2024. [Online]. Available: https://github.com/MohsenFayyaz89/T3D

[14] Z. Lan, Y. Zhu, A. G. Hauptmann, and S. Newsam, “Deep Local Video Feature for Action Recognition,” *IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, 2017.

[15] M. Jain, J. C. van Gemert, T. Mensink, and C. G. M Snoek, “Objects2action: Classifying and localizing actions without any video example,” *IEEE Xplore*, 2015.