Feature Exploration for Compelling Talks

Vaibhav Kalpesh Joshi
Department of Computer Science
Golisano College of Computing and Information Sciences
Rochester Institute of Technology
Rochester, NY 14623
vj3470@rit.edu

***Abstract*—Pose estimation is a technique that employs concepts of computer vision and deep learning algorithms to capture human poses and skeletal features in visual media content such as YouTube videos. Pose estimation can also help to model human behaviour with respect to YouTube videos. We can determine the influence of body gestures and poses on the popularity of a video. In this project, we focus on Ted Talk videos and develop a proof-of-concept model to classify the videos based on their likes / views and thereby determine the key body joints that contribute the most to the final output. We have collected 100 Ted Talk videos across a strict criteria and have generated the skeletal coordinates of the joints using a pose estimator called OpenPose. We have performed normalization on the pose data and divided the videos into small segments. We have assigned the videos to three popularity classes :- Low, Moderate and High based on their likes. We have initially implemented a straightforward baseline Random Forest Classifier. We have then used a slightly complex LSTM classification model which is more suitable for handling continuous, time-series data. Lastly, we have used a state-of-the-art Adaptive Graph CNN (AGCN) model as a final algorithm which gave the maximum accuracy among the three. We have achieved decent accuracy measures for all the models but leaving room for future improvement.**

***Index Terms*—OpenPose; Pose Estimation; LSTM; Graph CNN; Deep Learning; Random Forest; Classification; Regression**

I. Introduction

Pose estimation is a technique that employs concepts of computer vision and deep learning algorithms to capture human poses and skeletal features in visual media content. In the past, pose estimation was confined to constrained environments such surveillance systems. However, with the advancement of deep learning, it has forayed into more unconstrained and natural areas such as video games, sports and YouTube videos. Pose estimation is also a one of the key aspects in modeling human behavior. Human behavior with respect to any video may be influenced by a multitude of factors internal and external to the video such as the video topic, the content, human actions and poses with the video, visuals effects, etc. One of the ways to analyze human behavior is to examine the popularity of the video. In the context of a YouTube video, the number of likes, views and comments are reasonably reflective of the human behavior towards that video. A low number of likes / views may indicate a negative response towards the video and vice versa for high likes / views. Although there can be multiple factors influencing this behavior, we focus on the visual gestures, poses and expressions of humans.

This capstone project takes inspiration from idea that human behavior may be influenced from the video gestures and features and not just the content or the topic. Here, the focus is on Ted Talk videos available on YouTube. Using pose data of the speaker, we design a classification algorithm using various techniques such as Random Forest, Long-Short Term Memory (LSTM), and Adaptive Graph Convolutional Network (AGCN). We aim to classify the videos into classes such as videos with low popularity, moderate popularity and high popularity. Subsequently, via this classification, we attempt to determine the key or compelling joints and bones that contribute the most to a popularity of a video.

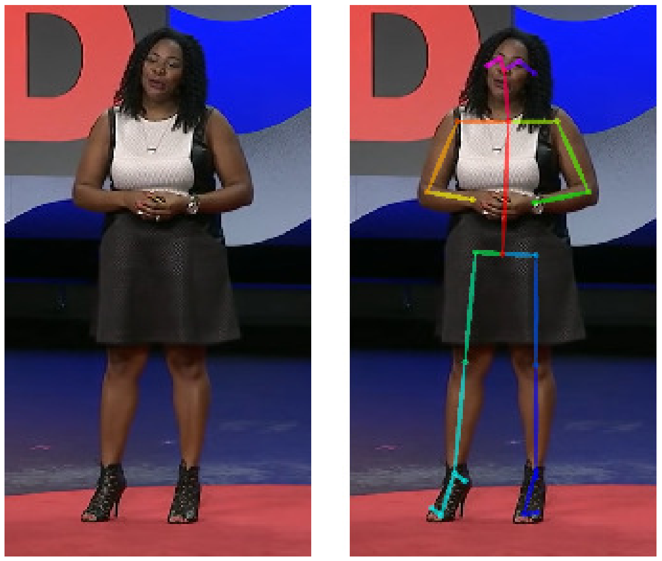


Fig. 1. OpenPose output for a frame in a Ted Talk video

In recent contemporary work, the use of pose estimation algorithms such as OpenPose [1] have been on the rise. These are openly available tools which conveniently generate skeletal coordinates of body joints from videos and still images. For obtaining the visual gestures from a Ted Talk video, we make use of OpenPose, which is an open source pose estimation library developed and maintained by Perceptual Computing Lab at Carnegie Mellon University [1]. It takes a video as an input and detects human body, hand, facial, and foot key

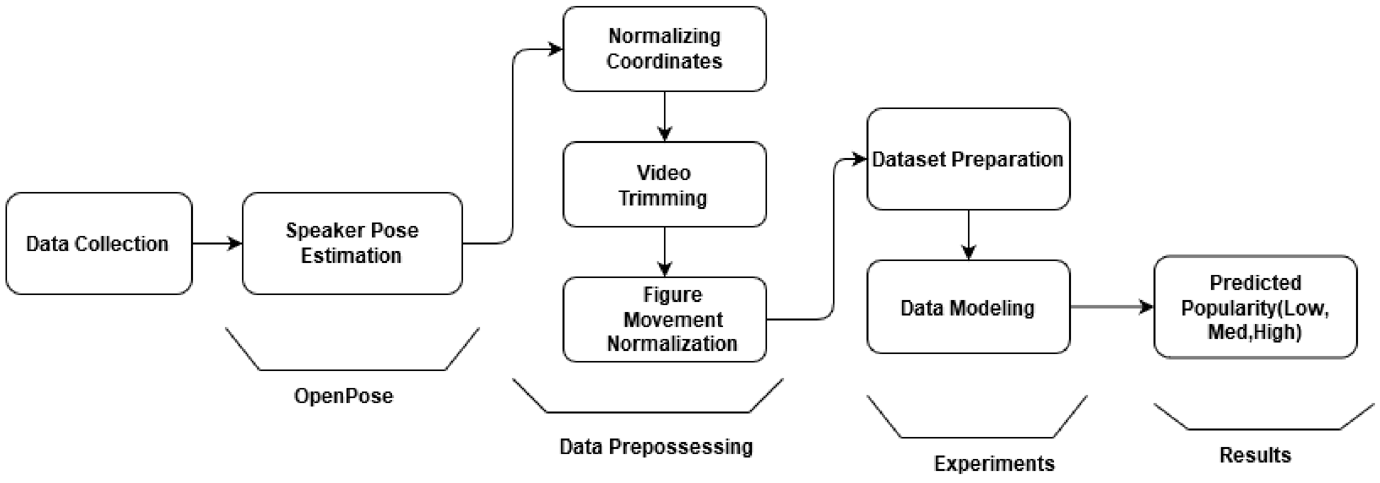


Fig. 2. System Architecture

points (in all 135 key points) on a per-frame basis. Figure 1 shows OpenPose in action. We use OpenPose as a blackbox that will generate coordinates of the various body key points for a given input video. These key points along with the video dataset will be modeled to test the correlation.

LSTM networks have become increasingly popular in work involving action recognition via time-series data [2] [3]. This is due to the clever mechanism of the via which it can remember information across large time gaps which makes it suitable to handle time-series data such as Openpose. However, recent studies conducted involving skeletal data action recognition using Graph Convolutional Networks have been shown to outperform the LSTM based networks [4] [5]. We utilize both of these algorithms in our work.

Overall, via this project, we aim to design a model which uses the spatial coordinates of the body joints and classifies the Youtube videos based on their likes/views. We collect 100 Ted Talk videos while keeping in mind constraints such as distribution of likes/views, age, duration, speaker sex, etc.. We use Openpose to generate the speaker pose coordinates for each of the videos. We perform normalization to counter for camera zooms and changes in the position of the speaker. We segment the videos in small chunks and create a large dataset to test our model. We design a baseline random forest to get a better idea for the more complex models. We design an LSTM based classification network to predict the popularity of the view. We leverage the state-of-the-art AGCN model for our analysis. We provide results in the form of confusion matrices and accuracy. We also perform analysis from the random forest results and try to identify the key features that affect the outcome of the classification. This preliminary analysis will set the groundwork for analyzing the more complex models (LSTM and AGCN) in the future.

The main aim of this project is to predict the popularity of the videos via skeletal data of body gestures and hence a low accuracy is merely an indicative of the failure of the hypotheses rather than the failure of the overall project. The work done in this project will serve as a proof-of-concept model for future iterations of the work to be carried on by prospective capstone students.

The rest of the paper is ordered as follows. Section II details the background and the related work. Section III describes the data collection procedure. Section IV details the methodology and the data models that are used. Section V discusses the various experiments carried out while Section VI discusses the results. Section VII analyses the results and Section VIII details the future work that needs to be done. Section IX provides the overall conclusion.

II. Related Work

There has been a fair amount of work involving feature identification, classification and action recognition using data derived from OpenPose. There have been numerous studies who have modeled OpenPose data to find compelling features across domains such as sports, YouTube videos, security, and health care.

LSTM has also been used increasingly in modeling time-series data. Moodley et al. [6] use LSTM and OpenPose to identify key features that contribute to a particular shot of a batsman in cricket. They classify the strokes of a batsman in three classes. Chen et al. [7] look at key features that lead to a person's fall. They also employ Openpose to extract the human skeleton and model the data via an LSTM and to recognize falling behaviour.

In the context of YouTube videos, work has been done to determine popularity of the videos using facial expressions of human beings [8]. However, a large chunk of work involving Youtube videos is based on action recognition. Kinect-Skeleton [9] is a large scale dataset consisting of snippets of Youtube videos across 400 actions. Yan et al. [5] have generated body skeletal joints of the raw videos in the dataset and released their data publicly. This data has been used by numerous works to predict the action of the person using LSTM [10], Graph Convolutional Networks [4] [5] and Feature Encoders [11].

Random Forest has also been used in extraction of key features from skeletal data. Xu et al. [12] use random forest to classify action of an online course instructor. They construct a feature vector from a video segment which captures the spatial and temporal characteristics of the instructors movement. We use the work done here while implementing Random Forest for our task.

Overall, there have been numerous works involving key feature identification, skeletal data modeling and YouTube video and they are all majorly related to action recognition. Although there is minimal work involving YouTube videos and their popularity, inherently all these tasks have classification at their core and hence it reasonably resembles our work.

III. Data

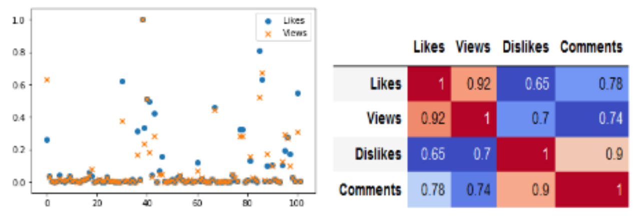
For this project, we chose to collect our dataset manually. Although there are popular dataset available for working with pose estimated data such as Kinect-Skeleton [4], they are mostly concerned with random snippets of YouTube videos and the videos are of considerably smaller length (around 10 seconds). Given that this is going to be a proof-of-concept model, we wanted the dataset to be large enough for useful analysis and not prone to overfitting. The following criteria were used to collect the videos:

A wide range of Likes/Views. This is important for regression problems since a fixed range of likes/views will be highly prone to overfitting. Also, our target variable will be either the likes or the views. Here, we chose likes over views but either one can be chosen since they are highly correlated as per Figure 3

Videos should be prior to 2017. This is needed to account for the maturity of a video as a more recent video might have fluctuation in the likes/views after some time which could cause instability in the data.

Maximum speaker visibility. In order to maximize the data points, we choose videos where the camera is focused on the speaker for most part of the video.

Equal Distribution of Male/Female Speakers. This is also to prevent overfitting and keeping our dataset robust.

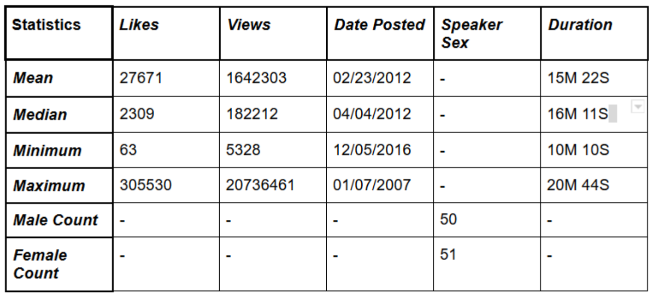


	Likes	Views	Dislikes	Comments
Likes	1	0.92	0.65	0.78
Views	0.92	1	0.7	0.74
Dislikes	0.65	0.7	1	0.9
Comments	0.78	0.74	0.9	1

Fig. 3. (L to R)-(a) Correlation between Likes and Views,(b) Matrix showing correlation between different attributes

Best resolution available. This is to maximize the data consistency across the dataset. Also, Openpose output will be more accurate with higher quality videos.

Figure 4 shows the overall statistics of few of attributes of the videos that were collected.



Statistics	Likes	Views	Date Posted	Speaker Sex	Duration
Mean	27671	1642303	02/23/2012	-	15M 22S
Median	2309	182212	04/04/2012	-	16M 11S
Minimum	63	5328	12/05/2016	-	10M 10S
Maximum	305530	20736461	01/07/2007	-	20M 44S
Male Count	-	-	-	50	-
Female Count	-	-	-	51	-

Fig. 4. Statistics of Attributes

IV. Models and Methodology

The Figure 2 shows the overall system architecture of our work. Our approach consists of the following steps :

A. *Speaker Pose Estimation*

To generate the spatial coordinates of the speaker's pose, we use OpenPose [2], which is a open source pose estimator developed and maintained by Carnegie Mellon University. It provides spatial coordinates for 135 features including the hand (40 points), pose (25 points) and facial features (70 points). Figure 5 shows the output spatial coordinates of the face, hands and the pose.

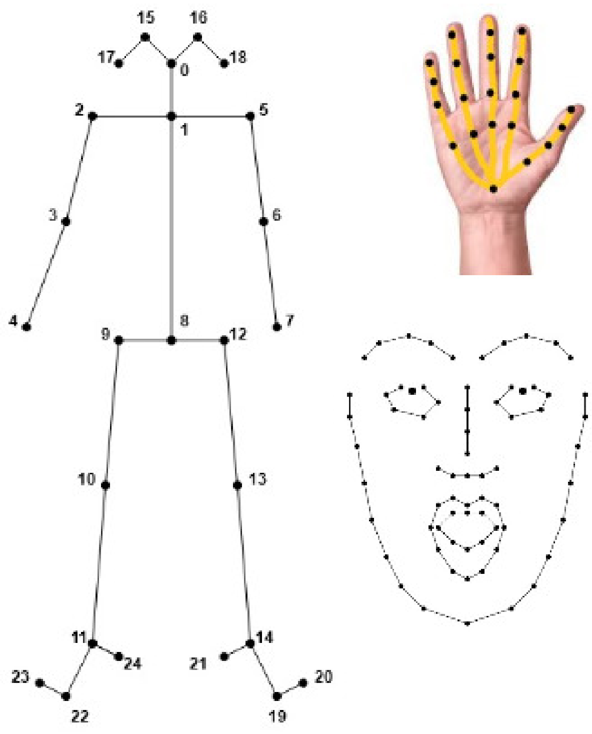


Fig. 5. (L to R clockwise) OpenPose coordinate key points - 5 a) Pose key points. 5 b) hand key points . 5 c) Face keypoints

The facial and hand coordinates are useful for more specific pose estimation tasks such as sign language detection and emotion detection. For Ted talks, the video is mainly focused on the entire speaker body and only occasionally focuses on the face. This leads to distortion and overlapping of points for the hand and face joints which causes Openpose to give inaccurate output or missing data. Also, for the purpose of our work, we need points that exhibit maximum variance across the video and facial and hand coordinates were not found to have large variance. Thus, for our work we have considered just the postural coordinates of the Openpose output.

In Figure 5, the 25 postural skeletal points that are generated by OpenPose are shown. Each point is associated with a body part. For instance, point 2 corresponds to the left shoulder while point 1 is the neck point. Upon observing the ted talk videos, we found that points 19 to 24 (leg joints) are not always visible and are often occluded in frames where they are present. Thus, we neglect those points. The remaining 19 points (0 to 18) are visible for a large part of the video and consists of key postural joints such as shoulders, arms, knees and elbows. Hence, we include these points in our analysis. This choice of points is also consistent with the data specifications of the AGCN model that we have used later. Openpose can also capture multiple people in a video but for the scope of our work we only focus on a single person.

B. Pose Data Normalization

We perform data normalization on the pose estimated data before feeding it into the model. The data generated by OpenPose differs across various video depending on their resolution, camera angles, etc. Hence, this is a key step and it sufficiently ensures that the data being fed to the model is consistent.

Normalizing Coordinates. OpenPose gives the skeletal coordinates with respect to the size (resolution)of the video. This can lead to discrepancies as older Ted Talk videos are generally in Standard Definition (480p) resolution and the newer videos are High Definition (720p) resolution. Openpose conveniently provides a argument flag that renders the coordinates of the videos in ranges of [0,1] and [-1,1]. For our task, we normalize the coordinates into a feature range of [0,1].

Video Trimming and Segmentation Although care is taken in collecting the videos that maximize the speaker presence, there are bound to be few frames that will not contain the speaker. For example, instances where the camera cuts to the audience, the speaker shows a presentation slide, etc. Thus, we discard those frames and concatenate the remaining segments into a single continuous video. Additionally, since we are focusing only on the postural or frontal-body data of the speaker, we eliminate the frames having a backward facing view of the speaker and also having a face-only (zoomed-in) view. Lastly, the initial starting and ending seconds of a ted talk video comprise of the credits which are not relevant to our analysis. Thus we trim the videos by a minute from the start and also the end.

Figure Normalization Our dataset contains Ted talk videos where the speaker and the camera are moving frequently and thus the relative positioning of the speaker in a video frame will vary from time to time. In a previous work done by Xu et al. [12], they attempt to normalize pose data of a course instructor via identifying points that are invariant with the position of the person. We use a similar approach to address the positional variation in our work. Ee use a specific pair of invariant joints from the speaker skeletal coordinates to normalize any distance-based feature values. We focus on the pair of points having the most presence in a Ted Talk video and exhibiting low variance relative to the whole body. We studied the postural data (Figure 5 (a)) and found that the point pair (1,8), i.e. neck and mid-hip joint are consistent throughout the video. We take the global average of this pair of points and use it as a normalization standard.

C. Dataset Preparation

Our dataset contains around 100 videos that are about 10-20 minutes in length. Choosing to use this data may have a few complications such as :-

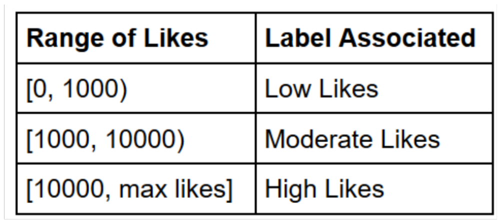
- **Insufficient Data for models :-**. Neural Networks tend to work well with large amounts of data. Hence, 100 videos split into 80 (train) and 20 (test) will not be suitable enough for a neural network to give good results.
- **Time Step issue for LSTM :-** LSTMs (explained in the next section) work by remembering information across certain time steps. Ideally, LSTMs should have under 1000 time steps for good performance. However, a 10-minute video has around 11000 frames and we consider each time step as one frame. Thus, we need to break the video into smaller chunks.

Thus, we address these complications by modifying the dataset before feeding it to the models. We perform segmentation of videos by splitting them into smaller chunks of 30 frames each. We then distribute the label i.e. the likes of that particular video to each of its segments. For instance, after segmenting videos of length 30, we effectively had 76000 videos with 30 frames each.

D. Data Modeling

Regression Model. Initially, the aim of our project was to develop a regression based model to predict the likes and views by identifying the compelling features that contribute the most to the popularity. Our reasoning being that likes/views are continuous values and thus more suitable to a regression task. However, after multiple experiments with models such as Random Forest, Vanilla LSTM and LSTM with Attention, the results did not seem to show a positive outlook. Also, the bulk of the contemporary work is classification oriented and hence we decided to tailor our project more towards a classification-based approach.

Classification Model. After achieving unconvincing results with regression, we chose to focus our task on classification of the videos based on their popularity. To achieve this, we split the videos into three classes as per the likes. As proven earlier, we choose likes over views but either can be chosen since they are highly correlated. Figure 6 shows the division and distribution of the classes for the respective videos.



Range of Likes	Label Associated
[0, 1000)	Low Likes
[1000, 10000)	Moderate Likes
[10000, max likes]	High Likes

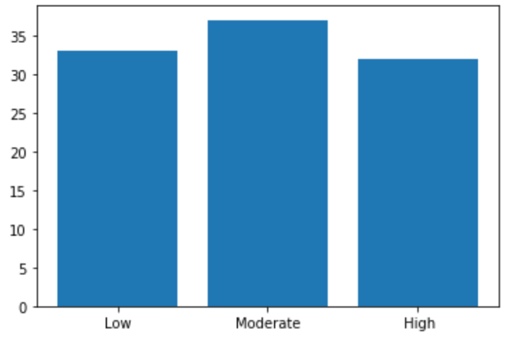
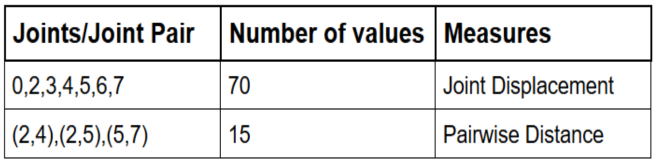


Fig. 6. Class Labels for Classification

1) Random Forest: Random Forest Algorithm has been used in tasks involving pose estimations and it often serves as a good baseline for more complex models. However, random forest are not suitable to work for multi dimensional data such as videos. Hence we need to convert multi-dimensional data into a single feature vector that best represents and captures the variances in the video. We have separated the videos into segments and hence we create one feature vector for each segment. We follow the work done by Xu et al. [12] and modify it to include points that are relevant for our analysis. Briefly, we use the joints 0,2,3,4,5,6,7 to calculate measures such as joint displacement, pairwise distance. Fig 7 shows the details.

Thus, we create a feature vector of 85 attributes for each segment of the video. Each feature vector has the likes of the video as its label.



Joints/Joint Pair	Number of values	Measures
0,2,3,4,5,6,7	70	Joint Displacement
(2,4),(2,5),(5,7)	15	Pairwise Distance

Fig. 7. Feature Vector creation for Random Forest

2) Long-Short Term Memory (LSTM) Network: LSTM is a kind of Recurrent Neural Networks (RNN) that has numerous applications in the fields of action recognition, time-series forecasting, speech generation, image captioning etc. It can persist information across multiple time steps owing to a loop and a clever execution of memory gates. It is specially designed to handle the vanishing gradient problem of traditional RNNs. This helps in remembering information that is useful in scenarios involving large time gaps such as those

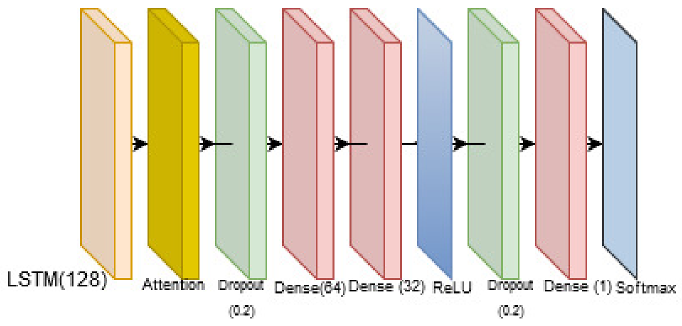


Fig. 8. LSTM Architecture

in video content. Additionally, LSTMs can be equipped with ”attention” mechanism to highlight or remember important features among a large number of features. This is important in our case as attention mechanism may be helpful in capturing the important or compelling body joints that contribute the maximum to the final result.

The architecture of our LSTM model shown in Figure 8 is explained below :

- **LSTM Layer :-** This layer contains the main LSTM network which takes in the input. The input of the LSTM is a 3D tensor with the following parameters :

  (batch_size,time_steps,input_dimensions)

  The batch_size is the amount of data samples per epoch. The time_steps parameter is essentially time ticks. It indicates how long in time each sample (video segment) is. For instance, a video segment of 30 frames is said to have 30 time steps. The neural network outputs one prediction after each time step. Finally the input dimensions are the number of features of the video segment. In our case we have 19 body points with a 2D coordinate pair associated with each which calculates to 38 dimensions.
- **Attention Layer :** This layer adds attention mechanism to output of the LSTM layer. We use the commonly used Luong style of attention for our task.
- **Dropout Layer :** These layers are added to handle potential over fitting issues. They essentially drop a subset of features at each epoch.
- **Dense Layer :** LSTM produces output as multi dimensional vectors. The dense or fully connected layers condense the LSTM output to a single value which is the prediction i.e. likes of the video.

  ReLU and Softmax layers are activation functions that are part of the Dense layers.

3) Adaptive Graph Convolutional Network (AGCN): Shi et al. [4] developed a state-of-the-art graphical neural network to address the task of action recognition for skeletal data. In graph CNN based action recognition, body skeletal coordinates generated by Openpose, etc. are represented as spatio-temporal graphs [4]. Graph CNN have been found to achieve high accuracy for action recognition compared to contemporary approaches such LSTM, RNN, CNN, etc.[5]

Kinetics-Skeleton dataset [9] is a massive human action dataset that comprises of 300,000 raw clips of Youtube videos across 400 classes of actions such as eating, drinking, clapping, etc. Yan et al. [5] use OpenPose to estimate the skeletal coordinates of each of the videos in the Kinetics dataset. They have released that data publicly which [4] use to test their AGCN model.

The work we are doing here is reasonably similar to the above since we too have Youtube videos, Openpose data and a video classification task. AGCN has outperformed the state of the art algorithms in the Kinetics dataset and hence seeing its potential we have decided to use it in our work. We largely use it as a black-box and just convert our Openpose data to their data specifications.

The overall architecture of the AGCN is shown in Figure 9. AGCN uses a two stream approach where the first order data (joint data) and the second order data (bone) are passed through two separate Graph CNN blocks and their results are ensembled together as a final prediction.

V. Experiments

We perform the segmentation of the videos as per the dataset preparation step and split whole videos into chunks of smaller segments. Each video is between 10-20 minutes in duration and processed at 25-30 FPS through the Openpose toolbox. For each of the model we make sure that we split the videos in such a way that segments from one video in the training set do not appear in the test set.

RandomForest. We use scikit-learn implementation of Random Forest Classifier. We initially apply 10-fold cross validation to the training set. We try out different parameters of the random forest algorithm such as the depth, number of trees and number of features and 64 trees to be the optimal value along with 20 maximum features. Different segment size chunks yield different dataset sizes. For instance, for segment size of 30 frames, we get 76000 total samples which we split into 61000 training and 16000 testing samples.

LSTM + Attention. The model summary of our LSTM network is shown in Figure 10. We do multiple runs and determine 128 as the optimal number of units for our LSTM. The input to our LSTM is a 3D Tensor of (32, 30,38) where 32 is the batch size, 30 is the number of time steps which is equivalent to the length of each segment (here 30). The input dimension correspond to the 19 (0 to 18) pose keypoints from the openpose data. Each point has an X-coordinate and a Y-coordinate thus totaling to 38 values. We have used Luong scheme for adding attention mechanism. We use the open source Tensorflow-Keras implementation of the LSTM model. We chose 'adam' as the optimizer and 'BinaryCrossEntropy' as the loss function.

Adaptive Graph Convolutional Network. We use the publicly available implementation of AGCN on Github for our analysis. To tailor our data to their code, we make changes to our openpose files to follow their input specification. We

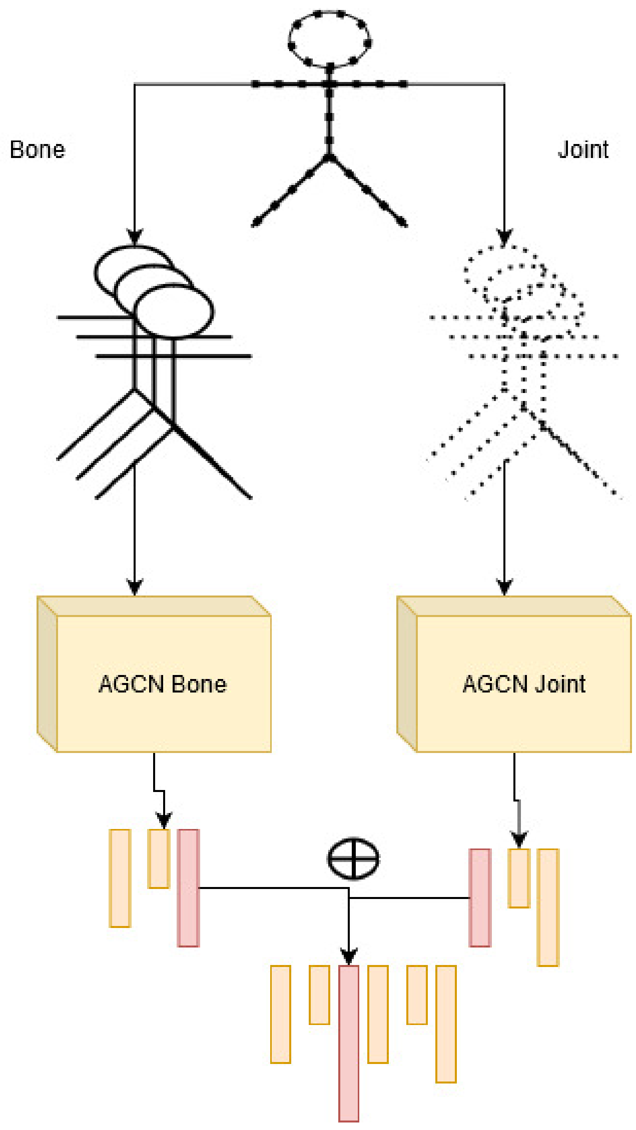
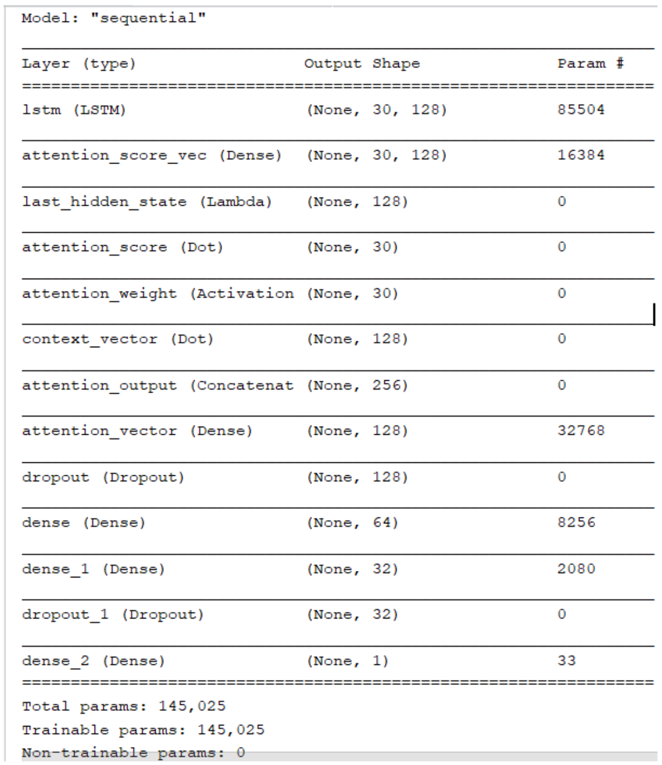


Fig. 9. AGCN architecture

make two separate folders for the training and testing set and a respective JSON file for storing the label. The folders store the openpose skeletal coordinates while the label information JSON file stores the ID of the video and its characteristics such as its label. The label file has a unique key for each video. Hence, we split our videos into segments and associate a unique key to each of the segments thus effectively treating each segment as a video itself. The label characteristics for each segment are inherited from the main video. We also make configurational changes to their code. For instance, AGCN uses 400 class labels while we have 3 main classes. We make this change and few others such as batch size, maximum frames per segment, etc.



```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm (LSTM)                  (None, 30, 128)           85504
_________________________________________________________________
attention_score_vec (Dense)  (None, 30, 128)           16384
_________________________________________________________________
last_hidden_state (Lambda)   (None, 128)               0
_________________________________________________________________
attention_score (Dot)        (None, 30)                0
_________________________________________________________________
attention_weight (Activation (None, 30)                0
_________________________________________________________________
context_vector (Dot)         (None, 128)               0
_________________________________________________________________
attention_output (Concatenat (None, 256)               0
_________________________________________________________________
attention_vector (Dense)     (None, 128)               32768
_________________________________________________________________
dropout (Dropout)            (None, 128)               0
_________________________________________________________________
dense (Dense)                (None, 64)                8256
_________________________________________________________________
dense_1 (Dense)              (None, 32)                2080
_________________________________________________________________
dropout_1 (Dropout)          (None, 32)                0
_________________________________________________________________
dense_2 (Dense)              (None, 1)                 33
=================================================================
Total params: 145,025
Trainable params: 145,025
Non-trainable params: 0
```

Fig. 10. LSTM Model Summary

VI. Results

A. Random Forest

We ran the random forest code for segment sizes of 30, 100 and 250. The confusion matrices are shown in Figure 11.

B. LSTM + Attention

We ran the LSTM implementation for segment sizes of 30, 100 and 200. The confusion matrices are shown in Figure 12.

C. Adaptive Graph CNN

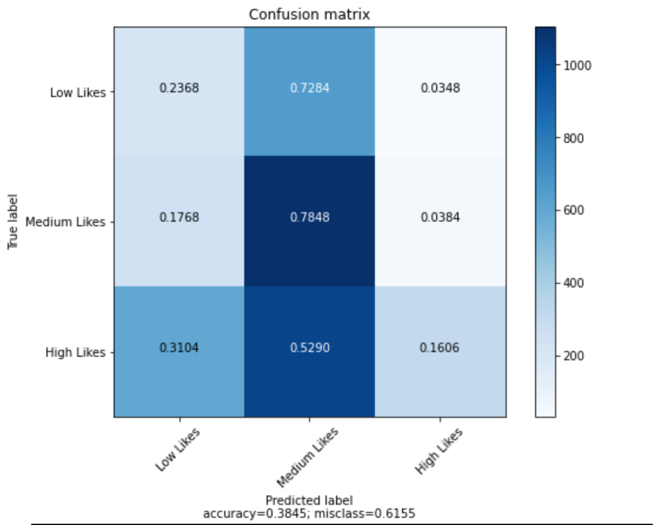
We used the publicly available code from the AGCN Github repository [13]. The implementation is in PyTorch. We use this code as a blackbox and generate outputs for segment size 30, 100 and the whole video itself. The table I shows the results of the AGCN implementation.

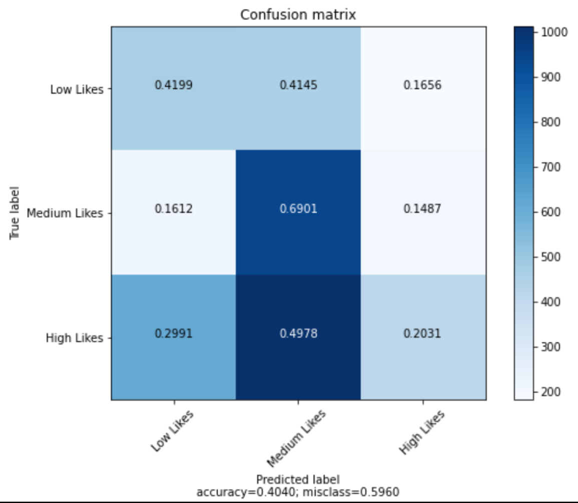
TABLE I
AGCN Accuracy Results

Segment Size	Sample Size	Accuracy Achieved
30	76319	43.13%
100	22369	44.11%
Entire Video	101	47.62%

VII. Analysis

Once we have built and tested the models, we perform preliminary analysis to determine which body features contribute the most to the popularity.





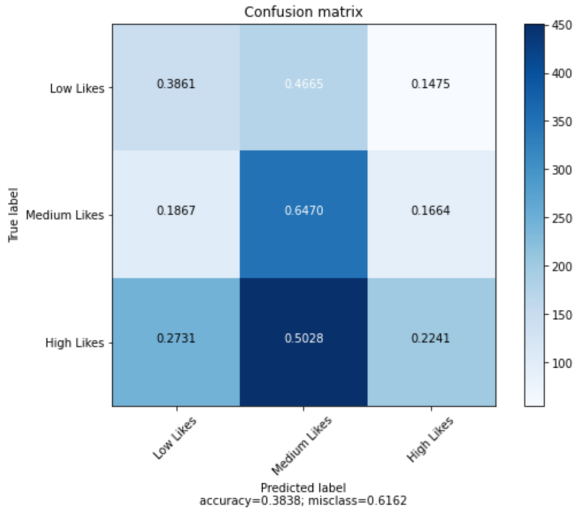
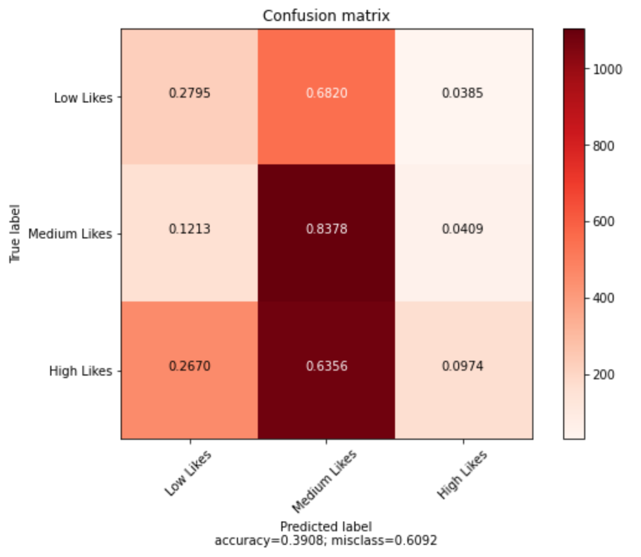


Fig. 11. (Top to Bottom) - a) Random Forest 30 Segments, b) Random Forest 100 Segments, c) Random Forest 250 Segments



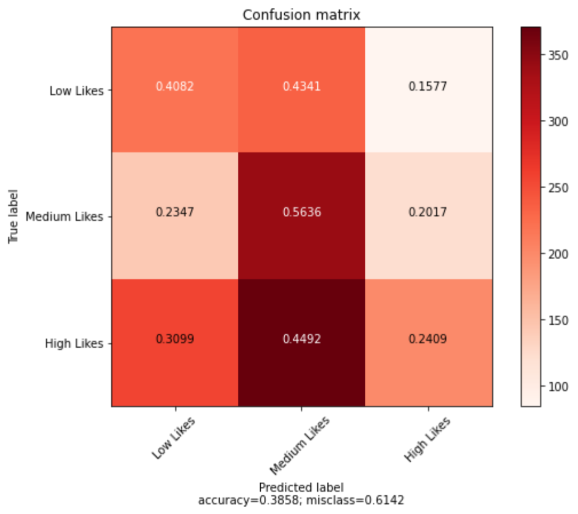


Fig. 12. (Top to Bottom) - a) LSTM - 30 Segments, b) LSTM 100 Segments

A. Random Forest

For the random forest model, we pre-select important postural feature points based on [12]. We experiment by adding new features and / or removing existing features and check on the accuracy. The analysis for the random forest algorithm is shown in the table II.

TABLE II
RANDOM FOREST FEATURE ANALYSIS

Joint / Joint Pair	**Accuracy After Removal**
2	35.11%
4	36.41%
5	35.62%
7	35.92%
(2,4)	34.03%
(2,5)	35.12%
(5,7)	36.42%
Removing all individual joints	31.13%
Removing all Pair joints	32.63 %

The pre-selected body joint key points for the Random Forest feature vector are 0 (Nose), 2 (Right Shoulder), 3 (Right Elbow), 4 (Right Wrist), 5 (Left Shoulder), 6 (Left Elbow) and 7 (Left Wrist). As per the analysis in the table II we find that removing a single individual points or a point pair has minimal effect on the accuracy. However, if all the individual points are removed and only the pair points or bones are considered, then there is a slight reduction in accuracy. Similarly, removing all the joint pairs also leads to a decent reduction in the accuracy.

This shows that feature vectors consisting of isolated pairs or individual points may not have as much influence on the popularity as opposed to feature vector consisting of pair-wise joints and individual joints.

Also, looking at the confusion matrices for various runs, the videos having higher likes have a higher misclassification rate compared to videos with low or medium likes.

This preliminary analysis serves as a baseline for performing detailed ablation study involving LSTM and AGCN models.

VIII. FUTURE WORK

Although we achieve decent accuracy numbers on our dataset and models, we leave room for improvement in some areas :-

Large Dataset. We only currently have 100 videos and thus we need to segment them into smaller segments to increase the data size. Smaller segments are not always beneficial in capturing a complete action or a speaker motion. Thus, having a large amount of videos, e.g. 1000 might enable us to segment into bigger chunks of continuous frames and yet have a high amount of data.

Ablation Study on LSTM and AGCN. Although LSTMs tend to perform really well for time-series data, the number of time-steps that they can retain the information is limited. LSTMs are found to perform well for upto 1000 time steps [14]. In our work, we correspond the number of time steps to the length of each video segment. This parameter can be experimented with in the future with a potential increase in the dataset and segmenting sizes.

We only perform process of identifying the key contributing features with the Random Forest model. For completeness of our work, we would need to perform an ablation study for LSTM and AGCN which may be similar to what is done with Random Forest. These models have the potential to give better results and capture more dynamism compared to Random Forest. We use the AGCN model by and large as a black box giving minimal attention to the internals. We might need to dive more into the internals of the best performing AGCN model and adjust the model as per our work requirements.

Transformer Networks. Transformer Neural Networks are a novel class of neural networks equipped with an Encoder-Decoder type architecture and a self-attention mechanism that offers improved computational performance and accuracy compared to CNNs and LSTMs [15]. LSTMs have a limited capacity of remembering information across a large time gap and while adding attention can mitigate that, they are still difficult to parallelize. Transformers were designed to address this limitation. Another interesting feature of transformers is that we can visualize what features the network attends on its way to computing a prediction [16]. In the context of our work, transformer networks can be helpful to observe which skeletal features the network takes into consideration while computing the popularity.

IX. Conclusion

Via this project, we have developed a baseline model to achieve classification of YouTube Ted Talk videos based on their skeletal points. We collected 100 videos and generated their postural coordinates via Openpose. We performed normalization, segmentation and implemented data models such as Random Forest, LSTM with Attention and Adaptive Graph CNN. We achieve accuracy in the range of 35-40% for Random Forest and LSTM. The state-of-the-art AGCN model achieved the best accuracy amongst all at 47.62%. We performed a basic post-analysis on the random forest results and found out that removal of individual body joints (out of a pre-selected feature range) does not affect the accuracy as compared to the removal to all individual joints. We also observe that videos having higher likes were misclassified more than videos having medium and low likes. We achieve decent results but there is room to improve this work, particularly the LSTM and AGCN models.

Acknowledgment

A special thanks to Dr. Ifeoma Nwogu not only who gave me the opportunity work on this project but also provided guidance and feedback throughout the project and without her invaluable insight this project wouldn't have been possible. I would also like to thank Fei Xu for valuable input during the project meetings, helping me with the coding questions and general guidance. Finally, I would like to my colloquium instructor Prof. Matthew Fluet for timely feedback on the presentation and documentation. I hope that Prof Ifeoma, Fei and anyone that carries this project forward in the future will benefit from my work. I will be happy to answer any questions and concerns that may arise in the future related to any aspect of my work.

References

[1] Z. Cao, T. Simon, S.-E. Wei, and Y. Sheikh, "Realtime multi-person 2d pose estimation using part affinity fields," in *CVPR*, 2017.

[2] A. P. YUNUS, N. C. SHIRAI, K. MORITA, and T. WAKABAYASHI, "Time series human motion prediction using rgb camera and openpose," *International Symposium on Affective Science and Engineering*, vol. ISASE2020, pp. 1–4, 2020.

[3] F. M. Noori, B. Wallace, M. Z. Uddin, and J. Torresen, "A robust human activity recognition approach using OpenPose, motion features, and deep recurrent neural network," in *Image Analysis*. Springer International Publishing, 2019, pp. 299–310. [Online]. Available: https://doi.org/10.1007/978-3-030-20205-7_25

[4] L. Shi, Y. Zhang, J. Cheng, and H. Lu, "Two-stream adaptive graph convolutional networks for skeleton-based action recognition," in *CVPR*, 2019.

[5] S. Yan, Y. Xiong, and D. Lin, "Spatial temporal graph convolutional networks for skeleton-based action recognition," *CoRR*, vol. abs/1801.07455, 2018. [Online]. Available: http://arxiv.org/abs/1801.07455

[6] T. Moodley and D. van der Haar, "CASRM: Cricket automation and stroke recognition model using OpenPose," in *Digital Human Modeling and Applications in Health, Safety, Ergonomics and Risk Management. Posture, Motion and Health*. Springer International Publishing, 2020, pp. 67–78. [Online]. Available: https://doi.org/10.1007/978-3-030-49904-4_5

[7] W. Chen, Z. Jiang, H. Guo, and X. Ni, "Fall detection based on key points of human-skeleton using openpose," *Symmetry*, vol. 12, no. 5, p. 744, 2020.

[8] P. Lewinski, "Don't look blank, happy, or sad: Patterns of facial expressions of speakers in banks' YouTube videos predict video's popularity over time." *Journal of Neuroscience, Psychology, and Economics*, vol. 8, no. 4, pp. 241–249, 2015. [Online]. Available: https://doi.org/10.1037/npe0000046

[9] W. Kay, J. Carreira, K. Simonyan, B. Zhang, C. Hillier, S. Vijayanarasimhan, F. Viola, T. Green, T. Back, P. Natsev, M. Suleyman, and A. Zisserman, "The kinetics human action video dataset," *CoRR*, vol. abs/1705.06950, 2017. [Online]. Available: http://arxiv.org/abs/1705.06950

[10] A. Shahroudy, J. Liu, T. Ng, and G. Wang, "Ntu rgb+d: A large scale dataset for 3d human activity analysis," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 1010–1019.

[11] B. Fernando, E. Gavves, J. M. Oramas, A. Ghodrati, and T. Tuytelaars, "Modeling video evolution for action recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[12] F. Xu, K. Davila, S. Setlur, and V. Govindaraju, "Content extraction from lecture video via speaker action classification based on pose information," in *2019 International Conference on Document Analysis and Recognition (ICDAR)*, Sep. 2019, pp. 1047–1054.

[13] L. Shi, Y. Zhang, J. Cheng, and H. LU, "Skeleton-Based Action Recognition with Multi-Stream Adaptive Graph Convolutional Networks," Dec. 2019. [Online]. Available: https://github.com/lshiwjx/2s-AGCN/tree/master/

[14] S. Li, W. Li, C. Cook, C. Zhu, and Y. Gao, "Independently recurrent neural network (indrnn): Building a longer and deeper rnn," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 5457–5466.

[15] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," *CoRR*, vol. abs/1706.03762, 2017. [Online]. Available: http://arxiv.org/abs/1706.03762

[16] J. Uszkoreit, "Transformer: A novel neural network architecture for language understanding," 2017.