StoryGraphs

Visualizing Character Interactions as a Timeline

Computer Blindness

Sreenya Chitluri - 2020102065
Sankeerthana Venugopal - 2020102008

Shot Boundaries

We use a normalized version of the DFD method to detect shot boundaries. This involves calculating the DFD for each pair of consecutive frames and then normalizing the result to account for differences in image size and intensity. By doing this, we can accurately identify the points where the motion between consecutive frames changes significantly, indicating a possible shot boundary. We assume that all scene boundaries only occur at shot boundaries, and so accurately detecting shot boundaries allows us to identify the boundaries between scenes in a video.

Shot Threading

- After identifying the shots of the video, we need to group the shots together to create some level of coherence.
- This is done by identifying similar shots by the following formula to check the distance between a set of shots -

$$\mathcal{C}(s, \mathcal{P}) = \frac{1}{|\mathcal{P}|} \sum_{p \in \mathcal{P}} \|H_s - H_p\|^2$$

- The similarity is computed between the last frame of each shot, and the first frame of the following 25 following shots.

Scene Detection

- We want to divide all the shots and scenes so that they form various scenes of the episode.

$$\mathcal{S}^* = \underset{S_i}{\operatorname{argmax}} \sum_{i=1}^{N_{sc}} \sum_{s \in S_i} \alpha(|\mathcal{P}|)(\phi(\mathcal{C}(s, \mathcal{P})) + T_{s,\mathcal{P}})$$

- The above formula is the brute force approach for the division of the scenes. This can be simplified into a dynamic programming problem as it can be converted into another problem of finding the optimal path from the base of the cube.

- Start at the last cell in the cube, which corresponds to dividing a sequence of length j into i segments using all available elements.
- From this cell, find the partition point k that results in the maximum value. This can be done by comparing the values stored in the cells corresponding to all possible partition points from 1 to j-1.
- Once the partition point k has been found, add it to the list of partition points that define the optimal partition.
- Repeat steps 2 and 3 for the subproblem corresponding to dividing the remaining part of the sequence (after the first partition point) into i-1 segments, using only the first k elements. This subproblem corresponds to the cell in the cube indexed by (i-1, k, 1).
- Continue this process recursively, each time finding the partition point that results in the maximum value for the current subproblem and adding it to the list of partition points defining the optimal partition, until all i-1 partition points have been found.
- The list of partition points obtained in step 5 defines the optimal partition of the j elements into i segments.

Face Detection

We need to know which characters appear in every scene to determine the positions of the character lines. To achieve this, we use a particle filter approach to detect and track faces in the video and extract DCT coefficients on aligned faces for each track. However, for some parts of the dataset, we do not have access to transcripts that would allow automatic person identification. To overcome this, we train multinomial logistic regression classifiers on 20% of the tracks for each character, including minor cast members. By using these classifiers, we can identify the characters present in each track and determine the positions of the character lines in the StoryGraphs.

Loss Functions

The four main components of the loss function are:

i) **Proximity -**

$$L_p^{(1)} = \frac{1}{N_P N_T} \sum_{c_i, c_j, t} p_{c_i, c_j, t} \cdot (x_t^{c_i} - x_t^{c_j})^2$$

$$L_p^{(2)} = \frac{1}{N_P N_T} \sum_{c_i, c_j, t} \mathbb{1}\{p_{c_i, c_j, t} = 0\} \cdot (x_t^{c_i} - x_t^{c_j})^2$$

$$L_p = L_p^{(1)} - L_p^{(2)}$$

Here, the x's represent the coordinate of the position in the graph of the character c at time t. The term p represents the normalized co-occurrence score. This is calculated by counting the number of frames of each character in a scene, and finding the geometric mean, and then finally thresholding using hysteresis thresholding.

ii) **Straight Lines -**

$$L_l = \frac{1}{N_C N_T} \sum_{c,t} (x_t^c - \mu_{\setminus x_t^c})^2 \qquad \mu_{\setminus x_t^c} = \frac{1}{N_T - 1} \sum_{q \neq t} x_q^c$$

This loss component ensures that the lines drawn on the graph are straight.

iii) **Minimum separation -**

Ensures that the lines do not overlap.

$$\mathcal{Z}(x, \mu) = \begin{cases} \frac{1}{x} \cdot (\sqrt{1 + (x - \mu)^2} - 1) & 0 < x < \mu \\ 0 & x \geq \mu. \end{cases}$$

$$L_s = \frac{1}{N_P N_T} \sum_{c_i, c_j, t} \mathcal{Z}((x_t^{c_i} - x_t^{c_j})^2, \mu_s)$$

iv) **Crossings -**

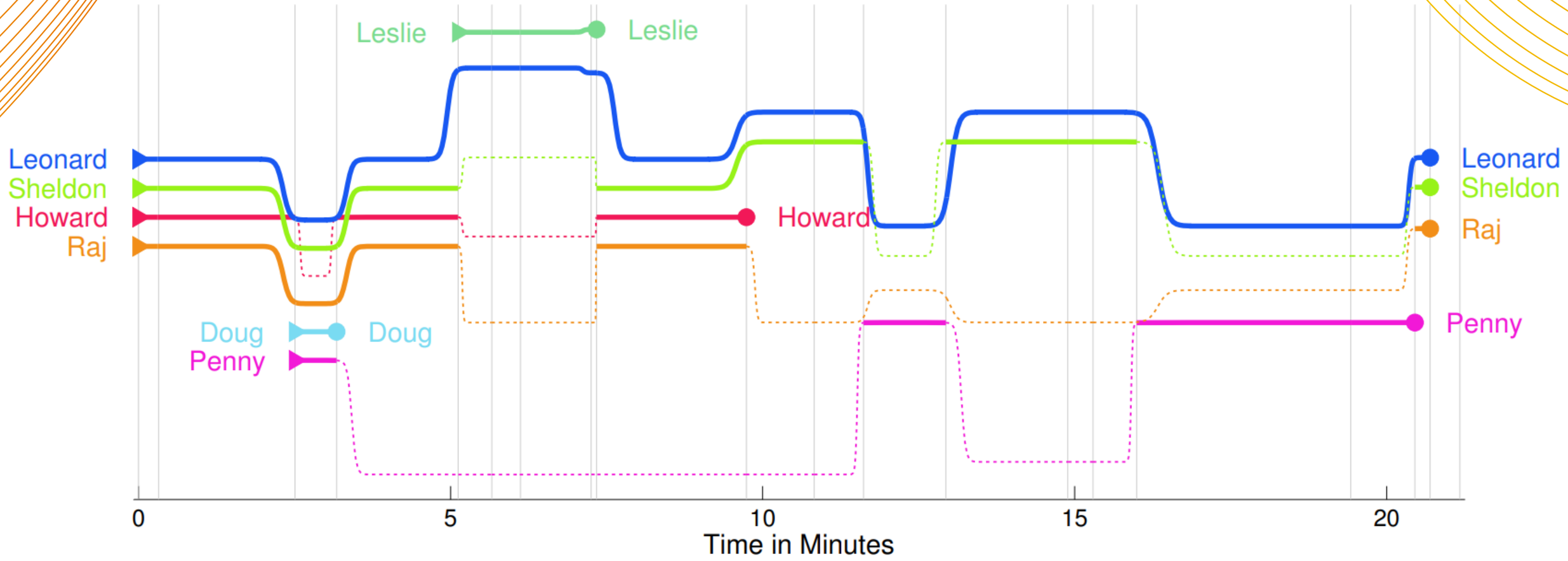
Tries to minimize the number of crossings of lines of different characters.

$$\mathcal{H}(x, \mu) = \begin{cases} \sqrt{1 + (x - \mu)^2} - 1 & x < \mu \\ 0 & x \geq \mu. \end{cases}$$

$$L_c = \frac{1}{N_P N_T} \sum_{c_i, c_j, t} \mathcal{H}((x_t^{c_i} - x_t^{c_j})(x_{t+1}^{c_i} - x_{t+1}^{c_j}), \mu_c)$$

The final loss is a weighted combination of the above losses.

Results


Leslie
Leslie
Leonard
Sheldon
Howard
Raj
Howard
Leonard
Sheldon
Raj
Doug
Doug
Penny
Penny
0
5
10
15
20
Time in Minutes

Thank You