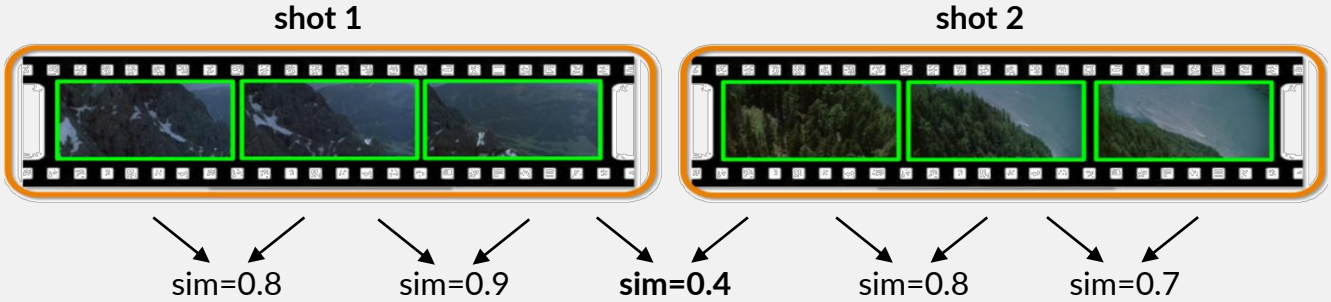
Task 1.4: Performance of Video Shot Detection (practical)

We developed software to segment videos into shots, which are sequences of frames that belong together. The software calculates frame-to-frame similarity and detects shot boundaries when the similarity falls below a set threshold, while frames with higher similarity are considered part of the same shot.



In the example above, we have two shots, each comprising 3 frames, with similarity values between consecutive frames. Notably, the similarity between frames at the shot boundaries is significantly lower than within the shots. By applying a threshold, such as 0.5, we can reliably identify the shot divisions.

Let's consider a practical example. You can download two text files on the course homepage, each representing the output of two different versions of our shot detection system. These files contain similarity values between consecutive frames, along with labels "shot" or "noshot" indicating the presence or absence of a shot boundary between the frames (these labels are used for system training). Preferably, use a Jupyter notebook (Python) and use markdown segments to document your answers and to explain your choices (minimal wording).

a) Begin by defining "positive" and "negative" for the actual condition observed in the text file as "shot" and "noshot." Similarly, establish what "True" and "False" mean for the predicted condition expressed as a predicate on similarity values between consecutive frames (and a chosen threshold).

b) Create a function to compute the confusion matrix's true positive (TP), true negative (TN), false positive (FP), and false negative (FN) values for a given threshold. Next, calculate the sensitivity (True Positive Rate or TPR), specificity (True Negative Rate or TNR), and accuracy (ACC) from these values.

c) Compute the ROC curve and create a plot using various thresholds. Determine the range of thresholds that is required. Explore whether there is a more efficient method for calculating the ROC curve rather than iteratively invoking the function created in step b) for the entire threshold value range.

d) Determine the ideal threshold for shot detection. What criterion are you using to define "optimal"? Evaluate the performance of both methods using their respective optimal thresholds.

e) Define a function that calculates the area under the ROC curve and compute it for both methods. How does the area relate to the values calculated in the previous sub tasks? Can you implement an efficient way to compute the area?