Video Key-Frame Extraction using Unsupervised Clustering and Mutual Comparison pdf용 요약

Video에서 key-frame을 추출한다.

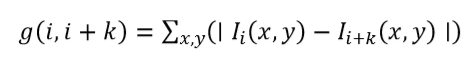
Unsupervised clustering 과 mutual comparison을 사용

Mutual comparison에는 color component와 texture를 사용한다.

Shot Boundary Detection (shot: video segmentation)

-video stream으로부터 visual features(color, shape, texture, motion) 또는 그 feature들의 결합을 뽑아내고 frame들 사이의 similarity를 측정

이때 similarity에는 g(i,i+k)사용, frame의 difference 또는 discontinuity를 나타냄



이때 I(X,Y)는 image pixel의 (x,y)좌표의 intensity level을 나타냄

하지만 위의 식은 intensity의 변화나 object, camera motion에 매우 민감 false 확률이 증가할 수 있음 (논문에서는 위에 식은 안 쓰고 밑에 나오는 것만 사용한 것 같음)

가장 선호되는 shot boundary detection 방법은 histogram-based method(frame의 color 분배)

Color histogram의 한계는 비슷한 histogram을 가진 이미지가 서로 다른 visual appearance를 가질 수도 있다.

key-Frame Extraction

* Shot segmentation 과정을 거치고 나면 shot의 frame들이 매우 비슷하기 때문에 이중 shot의 내용을 가장 잘 반영하는 key-frame을 뽑아냄.
* 크게 6가지의 분야가 있다(sequential comparison-based, Global comparison-based, Reference frame-based, Clustering, Curve simplification-based, object/events)

위의 분야 중 clustering 사용

Supervised clustering에서 shot boundary detection으로 얻어진 shot의 N개의 frame을 m개의 cluster로 분류

두 frame 사이의 similarity는 color와 texture요소의 가중치 조합 사용

Color feature은 HSV color space의 global level histogram사용

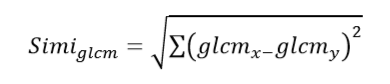
Texture feature은 GLCM을 사용했으며 GLCM에서 사용된 feature들은(Contrast, Correlation, Energy, Homogeneity)

Histogram similarity는 color bin들의 최소값의 합을 사용하는 histogram intersection method 사용하였으며 다음과 같음



Hx와 hy는 frame x와 y 각각의 HSV histogram, C는 histogram 안의 color bin 수

GLCM texture features를 이용한 frame similarity index는 다음과 같음 Euclidean distance method를 사용하며 다음과 같음



Glcmx와 glcmy는 각각 frame x,y의 GLCM texture feature

위의 두 요소 모두가 videao frame의 visual content를 표현하는데 똑같이 효과적인 것이 아니므로 가중치 필요(feature들의 중요성에 기반) 논문에서는 color histogram에 70%, GLCM texture에 30% 따라서 결합된 식은



위의 식을 이용하여 clustering 진행, 이때 clustering density를 제어하는 임계 값 δ 조절. δ가 높을수록 cluster가 많아짐. 새로 들어온 값에 대한 value가 δ보다 낮으면 이는 노드가 cluster에 추가되기에 충분하지 않다는 것을 나타냄. (이때 value는 node와 cluster 중심 사이의 유사성)

Clustering 과정

1. 초기화: f1을 cluster1에 넣고 f1을 cluster1의 중심으로 설정, numCluster = 1;
2. 다음 frame fi를 가져옴 이때 frame pool이 비어 있으면 6번으로 이동
3. Fi와 이미 존재하는 cluster(K = 1,2,…..,numCluster) 사이의 유사성을 비교: simi(fi,clusterK)
4. Maxsimi를 계산하여 어떤 cluster가 가장 가까운지 결정함

이때 Maxsimi 는 

Maxsimi가 임계 값 δ보다 작으면 충분히 가까운 클러스터가 없음: 5번으로 이동

그렇지 않으면 Maxsimi를 가진 cluster에 fi를 넣고 6번으로 이동

1. numCluster = numCluster + 1, 새로운 cluster에 fi를 넣음
2. cluster의 centroid를 조정, 이후 2번으로 이동

centroid 조정 = 

가장 왼쪽의 c0K는 새로운 centroid, 중간의 c0K’은 이전의 centroid, D는 cluster 안의 frame의 수

Cluster가 끝나면 key-frame을 선택해야함. 이 논문에서는 cluster 내부의 frame 수가 min\_clust\_size = shot의 전체 frame수의 10% 보다 크면 충분히 큰 cluster 라고 판단하며 해당 cluster의 key-frame만 선택. Cluster 내부의 key-frame은 cluster의 중심에 가장 가까운 frame을 선택. Min\_clust\_size를 줄이면 cluster의 수가 증가하고 over-segmentation 생길 수 있다. 반대의 경우 under-segmentation 생길 수 있음

논문 결과 over-segmentation의 결과로 추정되는 중복 프레임이 발견됨. Min\_clust\_size를 줄여보았지만 under-segmentation이 되는 한계 발견. 따라서 이를 mutual comparison을 사용하여 해결, key-frame을 다른 key-frame들과 비교하여 similarity가 특정한 임계 값보다 높으면 중복으로 판단 제거하였다.

그 결과 중복된 frame을 잘 제거함. 활동이 많은 video에서 더 많은 key-frame이 추출되고 활동이 적은 video 에서는 적은 key-frame이 나옴.

장점: easy to implement and fast to compute 무조건 첫 장면이 나오거나 하는 문제 없음, real time에 적용될 수 있음

Future work : deep convolution features로 해볼 예정