Video and Audio Data Extraction for Retrieval, Ranking and Recapitulation (VADER^3)

Volkmar Frinken[1(✉)], Satish Ravindran[2], Shriphani Palakodety[1], Guha Jayachandran[1], and Nilesh Powar[2]

[1] Onu Technology, San Jose, CA 95129, USA
{volkmar,spalakod,guha}@onai.com
[2] University of Dayton Research Institute, Dayton, OH 45469, USA
satishr@g.clemson.edu, Nilesh.Powar@udri.udayton.edu

Abstract. With advances in neural network architectures for computer vision and language processing, multiple modalities of a video can be used for complex content analysis. Here, we propose an architecture that combines visual, audio, and text data for video analytics. The model leverages six different modules: action recognition, voiceover detection, speech transcription, scene captioning, optical character recognition (OCR) and object recognition. The proposed integration mechanism combines the output of all the modules into a text-based data structure. We demonstrate our model's performance in two applications: a clustering module which groups a corpus of videos into labelled clusters based on their semantic similarity, and a ranking module which returns a ranked list of videos based on a keyword. Our analysis of the precision-recall graphs show that using a multi-modal approach offers an overall performance boost over any single modality.

Keywords: Multi modal video analytics · LSTM · CNN

1 Introduction

Recently, there has been considerable focus on trying to extract relevant information from video content, rather than just the metadata [2,8]. Understanding semantic content greatly improves access to video corpora through improved searching and ranking. Trying to extract relevant information using a single modality like the image or audio is prone to errors, either because of lack of accuracy of the processing algorithm or because of lack of underlying information in the modality under consideration. Fusing information from multiple modalities helps in providing more relevant results for video analytics. In this paper, we propose a novel way to integrate the information from a wide spectrum of information sources in a video. We will demonstrate our approach in two applications: ranking of videos in response to a search query, and clustering a corpus of videos based on semantic similarity.

L. Pancioni et al. (Eds.): ANNPR 2018, LNAI 11081, pp. 359–370, 2018.
https://doi.org/10.1007/978-3-319-99978-4_28

Even recent state-of-the-art techniques for video analytics either focus on extracting key frames from a video [7,15] or provide a textual summary of the video [10]. Since these approaches rely on visual information only and also focus on key subjects in the frame, they miss out on much of the contextual information that could be provided by the audio and background text.

2 Approach

Our approach addresses the shortcomings of the current state of the art by utilizing the information available in all the modalities in a video, i.e. the individual frames, audio and text. To our knowledge, this is the first time that a technique has been proposed which combines such a wide spectrum of information sources. Each of the independent modules operates on the input video after which the outputs from each module is combined into a text-based data structure.

We developed and tested the independent modules and will describe each of them in detail in this section.

2.1 Action Recognition

To recognize actions in videos, we combined deep learning based semantic segmentation approaches with recurrent neural networks. A high level overview of the bounding box detection network is given in Fig. 1. The first layers fulfill the function of semantic image segmentation. For this, we use the *DeepLap-MSc-COCO-LargeFOV* network provided by the University of California, Los Angeles [1]. Output activations from intermediate layers (as low level representation) as well as the pixel-wise output probabilities are fed into a long short-term memory (LSTM) layer [5]. The LSTM layer forms the recurrent part of the network and binds several frames together. The output of the network given a frame at time t therefore not only depends on the current frame, but also on previously read frames. At the top, a softmax output layer is used with cross-entropy training to recognize an action happening in the video frames.

For the textual representation, we divide the image into a pyramid: not only is the entire frame classified, but also the top-left, top-right, bottom-left, and bottom-right zoomed-in sum-frame, as seen in Fig. 2. Thus, each frame has five potential outputs, which are simply written in a line. If no action can be detected (the output activation of the *no-action* node is the largest activation), the output from that sub-frame is simply the empty string ε.

2.2 Voiceover Detection

The voiceover detection system is a neural network which evaluates whether the sound (such as voiceover text or music) in the video is added in a clean post-processing step or part of the original recording, captured at the same time (and with the same device) that recorded the video.

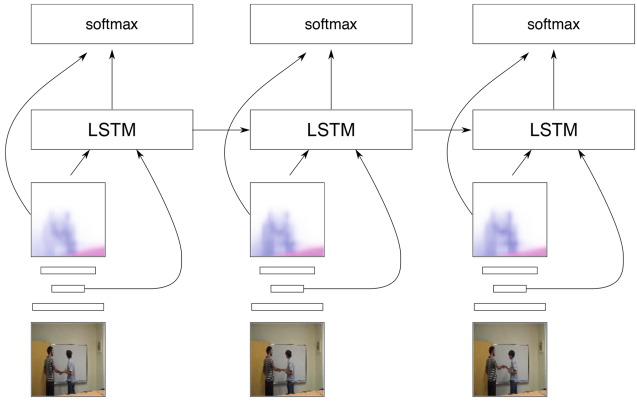


Fig. 1. High level overview of the action recognition neural network

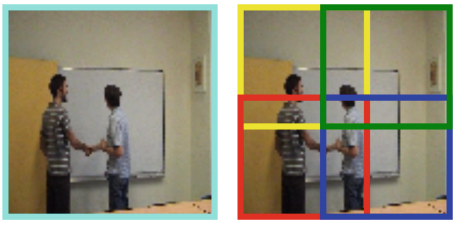


Fig. 2. The action recognition module is executed five times in parallel on each frame to cover actions at different scales.

The neural network designed to detect voiceover text is outlined in Fig. 3. Given the video's audio track, we extract a sequence of 13 Mel-frequency cepstral coefficients (MFCC) [14] with a frame rate of 25 ms frame width and a step size of 20ms. A larger step size than normally found in the literature allows for a faster processing and simpler model. Each group of 10 consecutive MFCC samples is recognized in a feed-forward neural network (with hidden layers of size 128, 64, 32, 16, 8, and 2) in a binary classification. The averaged binarized value is returned as the voiceover score, i.e., the fraction of time steps in which the *yes* output node has a larger activation than the *no* output node.

2.3 Speech Recognition

After comparison and research into the state of the art within the field, we settled upon using the Google Cloud Speech API[1]. From an input video, our module extracts the audio track and performs an API call to the Google cloud server. The length of the audio file accepted by Google is limited, so for longer audio transcriptions, we split the audio track into smaller segments with one second

[1] https://cloud.google.com/speech/.

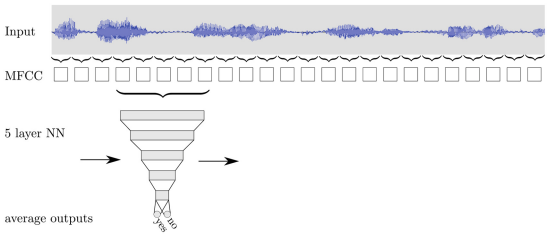


Fig. 3. A high-level overview of the voiceover detection system.

overlap. The returned transcription was directly used as textual representation, without any further processing. For more information about the Google Cloud Speech API, we refer to its documentation[2].

2.4 Automated Scene Captioning

Image and video captioning have seen much work in the past decade. As is typical in computer vision, an early emphasis was on still images rather than video. We used an encoder-decoder model where the image or video is first encoded into a semantically-rich embedding space and then the embedded representation is decoded using LSTM architectures. We leveraged the open source code associated with [16] for our application.

2.5 Text Detection and Recognition (OCR) and Object Recognition

For text detection, we initially tried a text specific object proposal algorithm described in [3]. Ultimately, we settled on using the OCR module in the Google Vision API[3] since it gave superior results.

For object recognition, we leverage the current state of the art CNNs to detect objects of interest in our database. We also evaluated other architectures including YOLO [12], DenseNet [6] and Resnet [4], but the Inception V3 architecture [13], released by Google performed much better in our tests.

2.6 Language Model Based Video Similarity

As explained in the Introduction, the previously introduced modules are run in parallel on an input video. Each of the modules returns a textual description of the different aspects of the video, such as speech, actions, objects, etc. The textual outputs are concatenated, cleaned, and normalized in the following manner: The URLs are first extracted and saved as words in the dictionary. The remaining text is transformed to lowercase. The Python NLTK word stemmer[4] is then

[2] https://cloud.google.com/docs/.
[3] https://cloud.google.com/vision/docs/ocr.
[4] http://www.nltk.org/api/nltk.stem.html.

applied to each word. We then save in a hash table, all word stemming transformation for a reverse lookup that is used later. Stop words from the NLTK stop word list[5] are removed. All the resulting words are then added to the dictionary. Finally, a token `<UKN>` symbolizing an unknown out-of-vocabulary word is added to the dictionary.

2.7 Video Ranking and Retrieval

All text documents created from the video database are represented as a bag-of-words. Similarities are computed using vector similarities between two frequency-inverted document frequency (tf-idf) [11] vectors of those bag-of-words. This provides a unified view for videos (which results in a matrix of pairwise distances) for arbitrary text queries. A query is transformed into a bag-of-words through the same steps outlined above. Words not occurring in the videos in the database are mapped onto the `<UKN>` word. Afterwards the vector similarities to all vectors in the database are computed and ranked. This provides a fast and robust method to retrieve videos that correspond to any arbitrary query. A sample demonstration is shown in Fig. 4.

2.8 Clustering

The pairwise video distances derived from the NLP-based text dissimilarities lend themselves well to hierarchical clustering, in our case agglomerative bottom-up clustering with single-linkage cluster distances. Starting from each video as a cluster of its own, a threshold is gradually increased (x-axis). As soon as that threshold is larger than the distance between two clusters, they merge into a new cluster, until finally all elements are part of one cluster.

The quality of the clustering is not easily measured by its own because it is not clear what a good cluster is without extensive ground truth. For the two main clusters, graduation speeches and TV commercials, we have an implicit ground truth given, but not at a finer level. Furthermore, there are ambiguous outliers. For example, consider a TV commercial with text in Spanish and a questions such as, "Are English language graduation speeches closer to English TV commercials than Spanish TV commercials to English TV commercials?" Since there is no clear answer to that, we jointly evaluate the clustering accuracy combined with the semantic cluster labels introduced next.

2.9 Semantic Labeling

After creating the clusters, we want to automatically generate cluster labels using the semantic information extracted from the individual modules. This is done using mutual information [9]. In a nutshell, considering the textual description of a video, we identify those words, whose occurrence (or lack thereof) serves best

[5] https://raw.githubusercontent.com/nltk/nltk_data/gh-pages/packages/corpora/stopwords.zip.

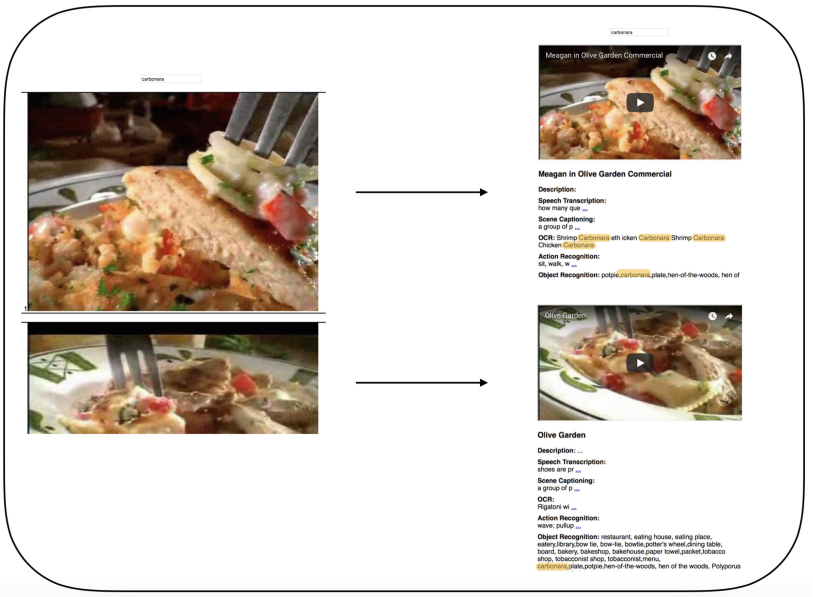


Fig. 4. The output for the query word "Carbonara". In the top video on the right the query word was detected by multiple modules (OCR and Object Recognition) resulting in a higher score. In the second video, the word was only detected by the Object recognition module.

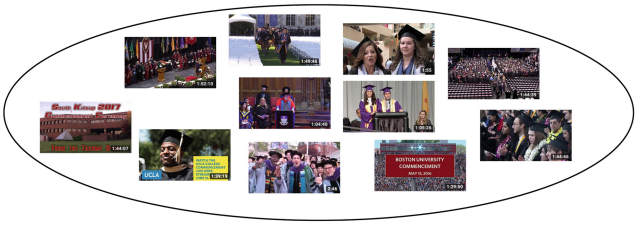
to predict whether or not a video is part of a cluster. Mathematically speaking, consider the mutual information of two random variables X, and Y:

$$I(X;Y) = \sum_{x \in X} \sum_{y \in Y} p(x,y) \log \frac{p_{X,Y}(x,y)}{p_X(x) p_Y(y)} \tag{1}$$

where $p_{X,Y}$ is the joint probability distribution of X and Y, and p_X and p_Y are the marginal probability distribution of X and Y. In our case, given a cluster C and a document D, X_C is a random variable to indicate membership in a cluster ($X_C = 1$) or not ($X_C = 0$), and Y_W indicates the occurrence of a word W ($Y_W = 1$) or the lack of it ($Y_W = 0$). Hence, $p(X_C)$ is the probability of a document being part of the cluster C, $p(Y_W = w)$ is the probability of a document containing the word W, and $p_{X_C,Y_W}(X_C, Y_W)$ is the probability of a document being a member (or not) of cluster C while containing the word W (or not). The mutual information becomes

$$I(X_C, Y_W) = \sum_{c=0,1} \sum_{d=0,1} p(X_C = c, Y_W = w) \log \frac{p_{X_C,Y_W}(X_C = c, Y_W = w)}{p_{X_C}(X_C = c) p(Y_W = w)} \tag{2}$$

The values $p_{X_C,Y_W}(X_C = 0, Y_W = 0)$, $p_{X_C,Y_W}(X_C = 0, Y_W = 1)$, $p_{X_C,Y_W}(X_C = 1, Y_W = 0)$, and $p_{X_C,Y_W}(X_C = 1, Y_W = 1)$ as well as $p(X_C = 0)$, $p(X_C = 1)$,



ceremony/ceremonial/ceremonies; graduates/graduated/graduate/graduation/graduating; colleg/college; nations/national/nation; sciences/science; next; commencement; ramp; school/schools; NOT commercials/commercial

Fig. 5. A cluster of videos and a ranked list of labels with the largest mutual information. Labels are word stems extracted during the pre-processing phase, which can result in multiple word instances as a label, e.g., ceremony, ceremonial, ceremonies.

$p(Y_W = 0)$, and $p(Y_W = 1)$ can be efficiently estimated for a given cluster and word either by counting the entire set or a randomly sampled set of document.

For each cluster, we consider the ten words with the highest mutual information. The mutual information is a measure by how much a cluster becomes predictable upon knowledge of the occurrence of a word. This is symmetric in both directions, where the existence or non existence of a word can provide information about the cluster. Therefore, we compute the mutual information between each cluster and word. Any word whose occurrence is negatively correlated with a cluster is appended with the prefix "NOT". Figure 5 shows a cluster of videos and the labels, and Fig. 6 shows all videos in a forced-directed graph, segmented into four clusters of at least two videos each, as well as a few singular videos.

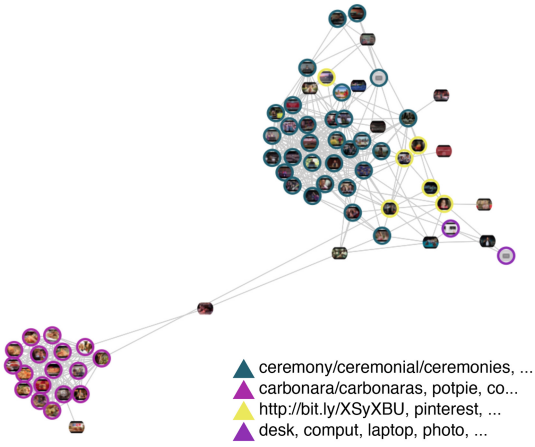


Fig. 6. An example where the threshold is set so that clusters arise with more than one video each.

3 Experimental Evaluation

3.1 Dataset

We manually annotated videos from YouTube belonging to two categories, *Commercials* and *Graduation Ceremonies*. The former consists of advertisements for phones, shoes, restaurants, and various other products or services. The ground truth included the object category, brand, a brief description of the activity and the text in the video. This category consists of 44 videos. The Graduation Ceremonies category consists of commencement ceremonies of various schools and colleges around the country. The labels includes the school name, the grade, date, and whether it is indoor or outdoor. We had 22 videos in this category. We selected them due to the readily available data and the diversity of content within these categories. The modules developed were tested on these datasets but can be applied to any video corpus.

3.2 Clustering and Labeling

We evaluated the cluster and the semantic labeling jointly using the following protocol. First, we took the list of the most important semantic keywords of all clusters, i.e., the list of all words that have the highest mutual information for at least 10 clusters. Those are the 53 labels shown in Fig. 7. For all videos in the database, we manually decided for each label whether it is an appropriate label or not. At times the labeling was ambiguous—for example, the label "room" can be seen in nearly all videos or "clock" is an object that may appear in the background in many videos. Also, for example, negative labels, such as "NOT loaf," are not easy to assign if a frame of an Olive Garden commercial shows a loaf of bread somewhere, yet the focus of the commercial is not the loaf. We handled all these ambiguities by letting the person annotating the video decide subjectively whether the label is appropriate or not. The large number of labels and commercials resulted in more than 3000 label decisions, and thus some inaccuracy in a few of the labels should not change the results significantly. In the next step, we created a rule-based system to decide whether a video should be part of a cluster or not. Given the list of keywords by the cluster labels, we consider for each video a binary vector of label relevancy. For example, a cluster might have the labels "olive/oliver, garden/gardens, mashed, consomme" then for an Olive Garden commercial focusing on pasta, the "olive/oliver" and "garden/gardens" labels are relevant, but not the "mashed" and "consomme" labels. Hence, the relevancy vector v would be $(1, 1, 0, 0)$. This needs to be reduced to a single yes/no-value to decide whether the video belongs to the cluster or not.

The *Min* rule assigned the minimal value $\min_i\{v_i\}$ to the relevance score of the video. In other words, a video is considered relevant under the *Min* rule, if all of the labels apply to the video. The *Median* rule assigned the rounded median value $\lfloor \text{median}\{v_i\} \rfloor$ to the relevance score of the video. In other words, a video is considered relevant under the *Median* rule, if at least half of the labels apply to the video. The *Max* rule assigned the value $\max_i\{v_i\}$ to the relevance score

carbonara/carbonaras	salad	room/rooms
olive/oliver	walk	riding/ride
garden/gardens	pour	NOT mashed
mashed	pinterest	NOT loaf
consomme	http://bit.ly/XSyXBU	NOT consomme
potpie	shoot	NOT burrito
potato	NOT potpie	live/living/lives
ceremony/ceremonial	colleg/college	http://bit.ly/XjG32m
cauliflower	breadsticks	history
pot	sciences/science	cradle
hotpot	NOT potato	channels
bench	lamp	cake
commencement	ramp	NOT pot
next	man	machine
nations/national/nation	kroc	clock
tennis	http://bit.ly/WUOGUw	universitys/universities/university
peopl	http://bit.ly/VFrhsx	/universe/universal
group	check	graduates/graduated/graduate/
frisbe	degree/degrees	graduation/graduating
president/preside	wrote	

Fig. 7. The labels that occur in the semantic labelling of at least 10 cluster. Note that the stemming joins different to the same stem, such as "graduat". Since "graduat" is not an English word, thus the returned label is the combination of all words mapped to it. The labels also include proper names, URLs, etc.

of the video. In other words, a video is considered relevant under the *Max* rule, if at least one of the labels applies to the video.

Figure 8 shows three recall-precision plots, for the three different rules. Each disk in the plot is one cluster. The *Min*, *Median*, and *Max* rule determine which videos should be part of the cluster. This is compared to the actual members of the cluster. From that we can compute the number of True Positives (TP), False Positives (FP), True Negatives (TN), and False Negatives (TN), which in turn is used to compute the precision of a cluster P and its recall R. Precision is a measure of a cluster's purity, the higher the precision, the less irrelevant videos are in the cluster. Recall gives the fraction of relevant videos being found. The larger the recall, the more videos that should be part of the cluster, are actually part of it.

The stricter the rule, the fewer videos should be member of a cluster. Sometimes, no video in the database should be part of a cluster, hence True Positive and False Negative must be 0, and the recall is undefined. In those cases, we do not plot any disk at all.

3.3 Individual Modules

In this subsection, we compare the performances of the individual modules with the combined analysis that take all modules into account.

Figure 9 shows three separate recall-precision plots for different cluster evaluation rules. A setting where half of the labels of a cluster must apply to video for it to be relevant appears similar to how a human user would evaluate correctness, but we include the extremes below for comparison.

A more detailed picture of the *Median* evaluation rule is shown in Fig. 10. Each circle represents a given cluster threshold. The size of the circle represents

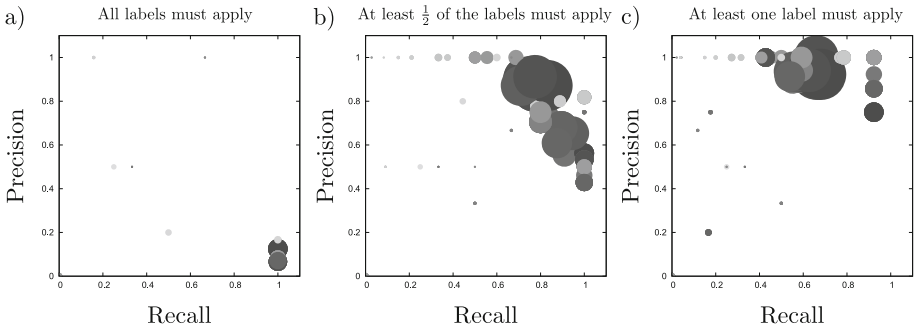


Fig. 8. Recall-Precision plots for all clusters given the three rules that determine whether a video should be part of a cluster. Each disk indicates one cluster, with the diameter of the disk indicating the size of the cluster while the color indicates the threshold.

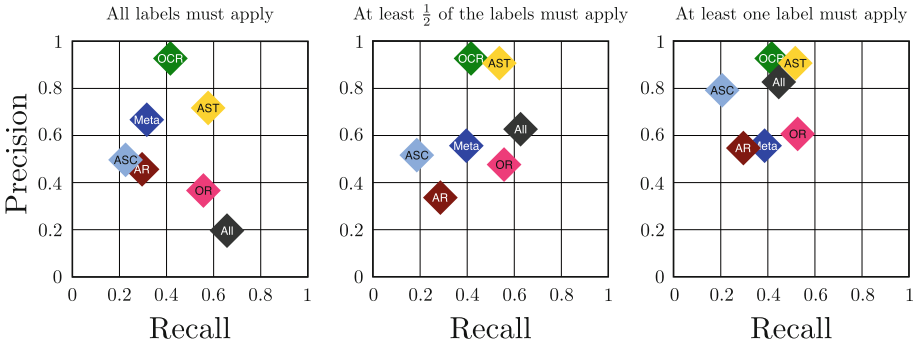


Fig. 9. Average precision and average recall values for the clusters generated using the individual modules (in color) and the combined system (grey). Evaluation of the cluster is done using the strict *Min* rule in the left plot, the more realistic *Median* rule in the central plot and using the relaxed *Max* rule in the right plot. (OCR = Optical Character Recognition, Meta = Video title and description, AST = Automatic Speech Transcription, ASC = Automatic Scene Captioning, AR = Action Recognition, OR = Object Recognition, All = Combined analysis) (Color figure online)

the number of clusters at that threshold. The black circles are from integrating together all the analyses. Note that all the black circles are towards the upper right corner, as desired (high precision and high recall). Certain individual analyses have high precision but fail to consistently accomplish both precision and recall.

For example, we can see that highly informative modules such as OCR return results with outstanding precision, yet they lack the power to find all videos, as can be seen by the comparatively low average recall value. Combining the modules gives a clear advantage as it finds more relevant videos, even at the cost of introducing some noise to the clusters.

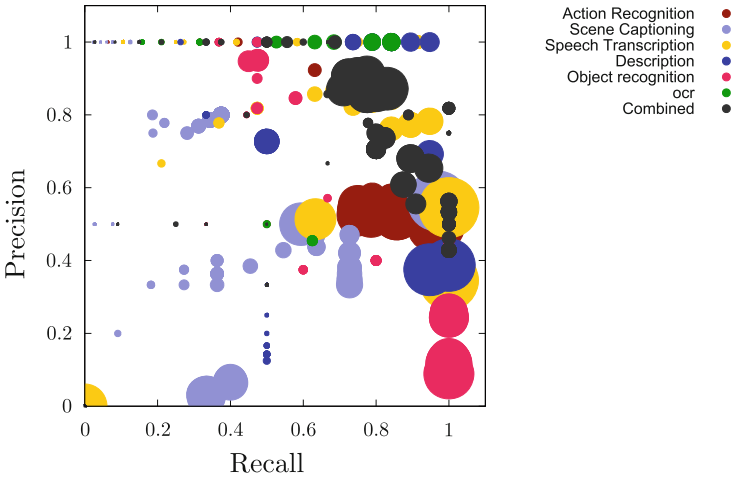


Fig. 10. Recall-Precision for all cluster created when considering only individual modules (in color) compared to a combined analysis (grey). Cluster ground truth is given by the *Median* decision rule. (Color figure online)

4 Conclusion

In this paper, we presented a mechanism for combining information from different modalities for video analytics. The visual, audio and textual information present in the video was converted into a combined text document. Latent Semantic Analysis was then used to compute a similarity metric between a corpus of documents, each document representing a video. We demonstrated two applications of our video analytics platform in this paper: (1) Video retrieval and ranking based on a keyword search and (2) Clustering of a corpus of videos based on semantic similarity of video contents. Our analysis show that combining the different modalities improves the overall robustness and performance of the system.

Acknowledgements. Supported by US Office of Naval Research

References

1. Chen, L., Papandreou, G., Kokkinos, I., Murphy, K., Yuille, A.: Semantic image segmentation with deep convolutional nets and fully connected CRFs. http://arxiv.org/abs/1412.7062 (2015)
2. Dong, J., Li, X., Lan, W., Huo, Y., Snoek, C.G.: Early embedding and late reranking for video captioning. In: Proceedings of the 2016 ACM on Multimedia Conference, pp. 1082–1086. ACM (2016)
3. Gómez, L., Karatzas, D.: Textproposals: a text-specific selective search algorithm for word spotting in the wild. Pattern Recognit. **70**, 60–74 (2017)

4. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. CoRR abs/1512.03385 (2015). http://arxiv.org/abs/1512.03385
5. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**, 1735–1780 (1997)
6. Huang, G., Liu, Z., Weinberger, K.Q.: Densely connected convolutional networks. CoRR abs/1608.06993 (2016). http://arxiv.org/abs/1608.06993
7. Ji, Z., Xiong, K., Pang, Y., Li, X.: Video summarization with attention-based encoder-decoder networks. CoRR abs/1708.09545 (2017). http://arxiv.org/abs/1708.09545
8. Kaufman, D., Levi, G., Hassner, T., Wolf, L.: Temporal tessellation: a unified approach for video analysis. In: The IEEE International Conference on Computer Vision (ICCV), vol. 8 (2017)
9. Manning, C.D., Raghavan, P., Schütze, H.: Introduction to Information Retrieval. Cambridge UP, Cambridge (2008)
10. Otani, M., Nakashima, Y., Rahtu, E., Heikkilä, J., Yokoya, N.: Video summarization using deep semantic features. In: Lai, S.-H., Lepetit, V., Nishino, K., Sato, Y. (eds.) ACCV 2016. LNCS, vol. 10115, pp. 361–377. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-54193-8_23
11. Ramos, J., et al.: Using TF-IDF to determine word relevance in document queries. In: Proceedings of the First Instructional Conference on Machine Learning, vol. 242, pp. 133–142 (2003)
12. Redmon, J., Farhadi, A.: Yolo9000: Better, faster, stronger. arXiv preprint arXiv:1612.08242 (2016)
13. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the inception architecture for computer vision. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2818–2826 (2016)
14. Xu, M., Duan, L.-Y., Cai, J., Chia, L.-T., Xu, C., Tian, Q.: HMM-based audio keyword generation. In: Aizawa, K., Nakamura, Y., Satoh, S. (eds.) PCM 2004. LNCS, vol. 3333, pp. 566–574. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-30543-9_71
15. Zhang, K., Chao, W.-L., Sha, F., Grauman, K.: Video summarization with long short-term memory. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9911, pp. 766–782. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46478-7_47
16. Zhou, L., Xu, C., Koch, P., Corso, J.J.: Image caption generation with text-conditional semantic attention. arXiv preprint arXiv:1606.04621 (2016)