A Unified, Modular and Multimodal Approach to Search and Hyperlinking Video

John Preston
jlp1g11@ecs.soton.ac.uk

Jonathon Hare
jsh2@ecs.soton.ac.uk

Sina Samangooei
ss@ecs.soton.ac.uk

Jamie Davies
jagd1g11@ecs.soton.ac.uk

Neha Jain
nj1g12@ecs.soton.ac.uk

David Dupplaw
dpd@ecs.soton.ac.uk

Electronics and Computer Science, University of Southampton, United Kingdom

ABSTRACT

This paper describes a modular architecture for searching and hyperlinking clips of TV programmes. The architecture aimed to unify the combination of features from different modalities through a common representation based on a set of probability density functions over the timeline of a programme. The core component of the system consisted of analysis of sections of transcripts based on a textual query. Results show that search is made worse by the addition of other components, whereas in hyperlinking precision is increased by the addition of visual features.

1. INTRODUCTION

The 2013 MediaEval search an hyperlinking task [1] tackles two problems; search across and within video collections, and hyperlinking of short video segments relevant to a given anchor segment. This paper describes the system we built to address the two tasks. The motivation behind the design of this system was to provide a uniform way of combining features across different modalities of data.

2. OVERALL APPROACH

The overall idea behind our approach was represent each programme by a probability density function (PDF) over the timeline of the programme. The area under the PDF between two time points essentially represents the probability of that portion of the programme to being relevant to the query. By constructing PDFs for each programme with a given query, we can then locate the high-probability segments of the PDFs, which in turn tell us the beginning and end times of hits that can be returned to the user. With respect to the search and hyperlinking tasks, the primary difference is the form of the query.

The architecture backing the approach was modular, and we constructed various modules to incorporate data from different data sources. We developed two forms of module; the first was capable of using the query to generate new data points for the PDFs for a set of programmes, and the second was capable of weighting the entire PDF for a given programme (i.e. to increase or decrease the global relevance). The modules responsible for adding to the PDF worked by placing Gaussian functions at the point of interest on the timeline, with variance proportional to the length of the segment of interest. At the end, the overall PDF for a programme can be computed from summation of the Gaussians at every time point; this entire process can be viewed as a variable bandwidth kernel density estimation.

Most of the work is done by modules working on the textual data (transcripts, synopsis, program titles). This information was indexed using Lucene[1] with separate fields for each source. Each module is described briefly below.

2.1 Generating Modules

The *transcript search module* was the most important component of the system, doing most of the heavy lifting when it came to determining the relevant sections of a programme. The module searched for keywords (taken from the query string) across all transcripts of a certain kind (LIMSI/Vocapia, LIUM, or subtitles) using the Lucene index. Keyword matches were extracted from each transcript in turn and grouped using hierarchical agglomerative clustering over the time differences between hits within a programme. When a cluster's separation (calculated as the temporal distance between the average values of the cluster's left and right children) fell below a specified threshold, a cluster was formed, and used to build a Gaussian whose amplitude was calculated from

$$\alpha = \frac{|W_Q|}{|Q|} \sum_{w \in W_Q} \text{boost}(w)\,\text{idf}(w)$$

where Q was the set of all possible keywords from the query, W_Q was the subset of query keywords that appeared in the transcript, $idf : W \to \mathbb{R}$ was a function mapping each keyword on to its inverse document frequency, and $boost : W \to \mathbb{R}$ was a function mapping each keyword on to its Lucene query boost. Additionally, the true amplitude was scaled by the normalised score returned by Lucene when searching for transcript documents matching the query. The amplitude of the Gaussian captures the relevance of all keywords in the cluster with respect to the document, as well as how completely the cluster covers the set of all possible query terms. The Gaussians were centred on the midpoint of the range covered by the cluster, and the standard deviation of the Gaussian was chosen as one third of the temporal size of the cluster plus 60 seconds.

The *concept module* analysed the query text and visual cues for known concepts that could be added to timelines. The amplitude for the concept module's Gaussians was de-

[1] `http://lucene.apache.org`

MediaEval 2013 Workshop, October 18-19, 2013, Barcelona, Spain

Table 1: Results for the search task

Run code	MRR	mGAP	MASP
S_M_Mod	0.208	0.0973	0.113
U_M_Mod	0.141	0.0812	0.0587
I_M_Mod	0.149	0.0828	0.0581
S_MV_ModCon	0.146	0.0743	0.0726
U_MV_ModCon	0.0808	0.0542	0.0401
I_MV_ModCon	0.0746	0.0412	0.0208
S_MV_ModConLSH	0.117	0.0652	0.0533
U_MV_ModConLSH	0.0510	0.0383	0.0211
I_MV_ModConLSH	0.0723	0.0431	0.0221

Table 2: Results for the hyperlinking task

Run Conf.	Retr.	P5	P10	P20	MAP
Subs	5393	0.42	0.35	0.22	0.069
Subs, cons	7489	0.35	0.30	0.20	0.059
Subs, cons, LSH	7488	0.35	0.30	0.21	0.059

termined from the normalised confidence for each concept detection, and the standard deviation was a constant 5 seconds.

The *visual information module* worked by finding shots that were visually similar to other shots with high confidence. For each programme, the most stable key-frame of each shot was extracted and SIFT features were calculated. Each SIFT feature was hashed using locality-sensitive hashing (LSH) and a graph was constructed where the vertices were keyframes and edges were created if pairs of key-frames contained colliding features [3]. The module found sections of timelines corresponding to shots whose integrals exceeded a threshold (i.e. shots already deemed relevant by the preceding modules), and added Gaussians centred on the shots whose keyframes were directly connected to this keyframe on the LSH graph. The base amplitude of the Gaussians was determined as the fraction of functions under which the two keyframes collided to the largest number of collisions. The amplitudes were additionally scaled by the integral of the shot from which the graph traversal originated. A constant width of 60 seconds was used. This module was implemented using OpenIMAJ [2] [2].

2.2 Weighting modules

The *synopsis* and *title* modules increased the weight of timelines belonging to programmes whose keywords matched keywords in the synopsis and title fields of the index. The *channel filter* module performed naïve NLP on the query: if it was found that a channel was mentioned in the query then any timelines corresponding to programmes on other channels were removed from the timeline set (i.e. setting the weight to zero).

3. SEARCHING AND HYPERLINKING

The architecture described in the previous section was used to facilitate both the search and hyperlinking tasks. For the search task, the system was configured to take the query and pass it directly to each module. Concepts were inferred from the query text and the visual information module was used for query expansion, through the detection of visually similar segments to high-confidence detections from the other modules. For hyperlinking, the transcript of the anchor segment was used as the query text (together with the synopsis, title and channel of the programme from which the anchor was drawn). Concepts detected in the anchor were used as input to the concept module. The visual information module was used to find segments that were visually similar to the anchor.

[2] `http://openimaj.org`

4. RESULTS

The results from the search task are summarised in Table 1. Runs using the subtitles gave the best performance in each category, which is understandable due to the more accurate nature of subtitles compared with speech-to-text transcripts. It is interesting to observe that the performance of the system decreased as additional features were brought in, which may indicate that these additional modules were not scaled properly, or otherwise very noisy in the context of the query. This is surprising for concept detection, as in the search task concepts were directly picked from the query.

Table 2 shows the results for the hyperlinking task. It can be seen that the baseline results with just the subtitle information used give the best results. Adding the concept detections causes a drop in performance; this indicates that similar visual concepts do not necessarily indicate relevant content. The addition of the SIFT-LSH features looks like it might slightly improve performance, but this needs to be verified with further experiments without the visual concepts.

5. CONCLUSION

The system performed better at the hyperlinking sub-task than the search sub-task; this was slightly unexpected as the search performance on the development data was higher. This may in part be due to fundamental limits to core aspects of the architecture, namely the transcript module: textual queries have a low bandwidth and describe many features that are not discernible from a programme's transcript, and thus a more complex approach might be required to improve performance. Additional NLP on queries, along with person detection (i.e. face recognition/verification), could also improve performance in the search domain, and we would like to explore this further in the future.

For both tasks the addition of visual information tended to harm overall performance. Again, this was slightly unexpected, as we saw improvements in the development search queries when using this information. One possible reason for this is that the visual information is only useful for certain types of query (or anchor). It would be interesting to explore this further in the future. A starting point for this would be to analyse the results on a per-item basis (rather than just looking at the overall averages).

6. ACKNOWLEDGMENTS

The described work was funded by the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreements 270239 (ARCOMEM), and 287863 (TrendMiner).

7. ADDITIONAL AUTHORS

Additional author: Paul H. Lewis (`phl@ecs.soton.ac.uk`)

8. REFERENCES

[1] M. Eskevich, G. J. Jones, S. Chen, R. Aly, and R. Ordelman. The Search and Hyperlinking Task at MediaEval 2013. In *MediaEval 2013 Workshop*, Barcelona, Spain, October 18-19 2013.

[2] J. S. Hare, S. Samangooei, and D. P. Dupplaw. OpenIMAJ and ImageTerrier: Java libraries and tools for scalable multimedia analysis and indexing of images. In *ACM MM'11*, pages 691–694. ACM, 2011.

[3] J. S. Hare, S. Samangooei, D. P. Dupplaw, and P. H. Lewis. Twitter's visual pulse. In *ICMR'13*, pages 297–298, New York, NY, USA, 2013. ACM.