Southampton WAIS and the MediaEval 2013 Search and Hyperlinking Task

Jonathon S. Hare
jsh2@ecs.soton.ac.uk

Sina Samangooei
ss@ecs.soton.ac.uk

David P. Dupplaw
dpd@ecs.soton.ac.uk

Paul H. Lewis
phl@ecs.soton.ac.uk

Electronics and Computer Science, University of Southampton, United Kingdom

ABSTRACT

Categories and Subject Descriptors

H.4 [**Information Systems Applications**]: Miscellaneous; D.2.8 [**Software Engineering**]: Metrics—*complexity measures, performance measures*

General Terms

Keywords

search, video, transcripts, agglomerative clustering, SIFT, locality-sensitive hashing

1. INTRODUCTION

A team from the University of Southampton's Web and Internet Science research group participated in the 2013 MediaEval Search and Hyperlinking challenge. A system was built to address the two tasks of retrieving video segments from an archive of programmes, given a textual query or an anchor segment from which to originate.[1]

2. ARCHITECTURE

For the search sub-task, the system was to take a query and return a series of clips as results. This necessitated extracting relevant or otherwise appropriate sections from programmes. To facilitate this, individual programmes were generalised to functions of interest over time, where the real value at a given time indicated the instantaneous relevance of that point within the context of the current query. This paradigm permitted a modular approach to the system architecture, where individual modules can be applied to add, remove, or modify a set of timelines.

The modules developed operated by placing Gaussians on a timeline or by adjusting the factor by which a timeline was scaled (so indicating an overall increase in the relevance of one timeline compared to another). Where a Gaussian was to be defined or a scale factor increased, the value was calculated as

$$a = S_m \alpha^{P_m}$$

where S_m was the scale factor for a module m, α was the module's base value for the operation, and P_m was the power factor for said module. This was done to provide the opportunity to control how much one module contributed to describing (some section of) a timeline as interesting, as well as the distribution of values created by said module.

2.1 Transcript module

The transcript 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, LIUM, or subtitles). Matches were extracted from each transcript in turn, and a binary tree of these hits was then built using agglomerative clustering. The tree was walked, and when a cluster's separation (calculated as the distance between the average values of the cluster's left and right children) fell below a specified threshold, the cluster was used to build a Gaussian whose amplitude was calculated from

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

where W was the set of keywords in the transcript, Q was the set of all possible keywords from the query, $idf : W \to \mathbb{R}$ 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 boost in the query, if any. Additionally, the true amplitude was scaled by the normalised score returned by the search engine when searching for transcript documents matching the query. Thus the amplitude of the Gaussian captures the relevancy 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 centered on the midpoint of the range covered by the cluster, and the parameter c of the Gaussian was chosen as one third of the temporal size of the cluster plus 60 seconds.

Originally a separate Gaussian was created for each keyword that matched in the transcript, but it was found that by examining clusters of keywords and taking into account how well each cluster covered the query, the quality of the results was improved.

2.2 Other modules

The synopsis and title modules increased the scale factor of any timelines whose synopses or titles matched the query by an amount derived from the search engine's score for the query in those fields and for the programmes corresponding

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

to those timelines.

A channel filter module was implemented which performed some 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.

A concepts module looked in the query text and visual cues for known concept detections that could be added to timelines. The amplitude for the concept module's Gaussians was determined from the normalised confidence for each concept detection, and the width was a constant 5 seconds.

Additionally, another module worked purely visually, finding shots that were visually similar to existing shots with high confidence. For each programme, the most stable keyframe of each shot was extracted and SIFT features were calculated. These features were clustered using locality-sensitive hashing (henceforth LSH) and a graph was built whose vertices were keyframes such that any two keyframes were connected if the number of functions they collide under exceeded a given threshold. The module operated by finding sections of timelines corresponding to shots whose integrals exceeded a threshold (i.e. shots already deemed relevant by the preceeding 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, and the true amplitudes were additionally scaled by the integral of the shot from which the graph traversal originated. A constant width of 60 seconds was used.

2.3 Hyperlinking

The hyperlinking sub-task was viewed as an extension of the search sub-task, where the search query was an anchor as opposed to a string. Thus hyperlinking was addressed by slightly modifying the search architecture.

A module was written to construct textual queries from anchors by extracting the portion of a transcript throughout the anchor segment. This allowed the use of the transcript, synopsis, title, and channel filter modules from the searcher architecture without any modification.

The concept module was re-written to find any concept detections during the anchor segment and to bring in other sections of video matching these concepts, whereas in the searcher architecture concepts were inferred from the query text.

The LSH SIFT graph module was re-written to operate by searching for similar frames starting with the frames in the anchor segment, whereas before this module was purely expansionary, operating on timeline sections that were already of high confidence.

2.4 Tools and techniques

In order to facilitate searching of transcripts, synopses, and programme titles, these features were indexed using Apache Lucene.

The Apache Commons Math library provided implementations of Gaussian functions and integrators which were used for constructing timelines and assessing the confidence of various segments, respectively.

Thomas Jungblut's implementation of agglomerative clustering (`https://github.com/thomasjungblut/tjungblut-math`) was used to enable clustering of transcript search hits.

SIFT features were extracted from keyframes using the OpenIMAJ library[2] and Apache Hadoop.

3. RESULTS

4. PROBLEMS ENCOUNTERED

A number of issues were encountered while developing the system.

The ground truth provided was not to a standard acceptable to facilitate automation of certain aspects of training of the system. Sometimes, the expected result was less relevant than other results which may be returned by the system. This was compounded by the fact that some queries were rather vague, and a number of results would adequately answer such queries. These factors placed a ceiling on the performance of the system, at least insofar as measurable by a metric based on sub-optimal ground truths.

The variety of different types of programmes eliminated the possibility of a single good all-round configuration for the system. For example, in a news programme keywords may pop up at regular intervals when headlines are read out, despite there being only a single short segment of said programme that contains the content the user is looking for. In contrast, a documentary may have multiple longer segments that are relevant, again with a different distribution of keywords. Ideally there would be some way to detect what genre of programme the system is extracting sections from, so as to maximise the potential relevance of such sections in the context of the query.

Originally sections were extracted from transcripts using Lucene's highlighting feature, but this was found to produce sub-optimal results: the highlighter tended to skew results towards the start of a transcript, and the length of the fragments returned was unpredictable, often extending much after the last keyword seen in said fragment. A manual approach to fragmentation using agglomerative clustering was used instead.

Additionally, there were a few issues with the provided data: typos in queries, unparseable subtitles, and missing concept detections, however these did not pose a systematic challenge.

5. FURTHER WORK

One feature that was not implemented was person detection: ideally the system would be able to extract names from queries or faces/names from anchor segments and match these via a pre-computed database mapping between face detections and names. A good approximation could have been built using the provided metadata on characters and actors in each programme, and a system was envisioned that would allow arbitrary names to be queried against an online image datasource such as an image search engine, from which a detector could be trained and used to process the dataset live.

As previously mentioned, the performance of the system may be improved by categorising programmes and developing profiles for modules based upon these classifications.

6. CONCLUSION

7. 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 *Proceedings of ACM Multimedia 2011*, MM '11, pages 691–694. ACM, 2011.