Exploring Microblog Activity for the Prediction of Hyperlink Anchors in Television Broadcasts

Raynor Vliegendhart
R.Vliegendhart@tudelft.nl

Cynthia C. S. Liem
C.C.S.Liem@tudelft.nl

Martha Larson
M.A.Larson@tudelft.nl

Multimedia Computing Group, Delft University of Technology, Delft, The Netherlands

ABSTRACT

In this paper, we present a social media based approach to finding anchors in video archives. We use social activity on Twitter to find topics on which people have questions about in order to select suitable anchors. The experiments were carried out on the MediaEval Search and Anchoring in Video Archives Task (SAVA) data set, consisting of 68 hours of BBC video content broadcasted in 2008. The performance of our relatively simple, but straightforward method seems sufficiently promising to pursue further research.

1. INTRODUCTION

One of the research questions of the SAVA task, and the one that is being addressed in this paper, is how to automatically identify anchors for a given set of videos, where anchors are media fragments for which users could require additional information [2].

Microblog platforms, such as Twitter,[1] reflect social activity that takes place around a TV show at the time that it is broadcast. Our approach is based on the idea that users will want to learn further information on segments that discuss topics that trigger questions. We use activity on Twitter to find which topics trigger user questions. The more Twitter questions associated to topics discussed in a certain shot, the greater we consider the likelihood that the corresponding part of the video represents a viable anchor. Our approach is further based on keyphrase mining. We understand keyphrases in the sense of [3], namely, as noun phrases that capture the main content of a document. We make the simplifying assumption that the relationship between questions and shots is reflected in the number of keywords that they share in common.

2. METHOD

Our approach to anchor generation in broadcast videos exploits social chatter about topics on the microblogging platform Twitter. The method requires that subtitles and shot boundary information for a video are available. Anchors for a given video are then generated as follows.

For each subtitle $s = (l, t_s, t_e) \in S$ consisting of a line of text l, a start time t_s, and an end time t_e, we extract set of keyphrases K_s using `nltk` [1] and a chunker from [3]. The set of all keyphrases is then denoted as $K = \bigcup_{s \in S} K_s$.

For each shot defined by its boundaries $b = (b_s, b_e) \in B$, consisting of a start time b_s and end time b_e, we introduce the notion of a subset $S_b \subseteq S$ representing all subtitles that start in that shot: $S_b = \{s \mid s = (l, t_s, t_e) \in S \wedge t_s \in [b_s, b_e)\}$. With these definitions set in place, we can now define all keyphrases that occur in a shot b:

$$K_b = \{k \mid s \in S_b \wedge k \in K_s\}\,.$$

To determine the importance of a keyphrase term $k \in K$, we retrieve tweets containing a question about the keyphrase by sending the following query to Twitter: "k `? since:`d_s `until:`d_e". In this paper, we use a fixed date range corresponding to the given set of videos for all keyphrases, regardless of the airing date of the video. This means we retrieve questions of social relevance during that general time period, not "the issues of today" in the past. Let $q : K \to \mathbb{N}$ denote the function to count the number of tweets retrieved for a keyphrase $k \in K$. The weight of each keyphrase k is then determined by a weighing function $w : K \to \mathbb{R}$. This function w is of the form $w(k) = f(q(k))$ where f is implementation-dependent and its purpose is to scale the tweet count returned by q.

We then rank shots by the summed weight of all keyphrases appearing in each shot. Let $W : B \to \mathbb{R}$ denote this summation: $W(b) = \sum_{k \in K_b} w(k)$. Then $r : B \to \mathbb{N}$ denotes the function that assigns a rank to a shot defined by its boundaries $b \in B$:

$$r(b) = 1 + \left|\{b' \mid b' \in B \wedge W(b') > W(b)\}\right|.$$

After ranking the shots, we simply generate anchors of an arbitrarily chosen minimum fixed length of $T = 30$ seconds, using shot boundaries for alignment. The underlying assumption is that cutting at shot boundaries should result in clean media fragments. Let $a : B \to \mathbb{R}^2$ denote the function that computes the start and time of the anchor derived from a shot $b \in B$. The start time is equal to the start time of the shot itself, i.e., b_s, and the end time is equal to end time of the first shot that ends at least $T = 30$ seconds later, or the end time of the whole video:

$$a(b) = (b_s, \min \begin{pmatrix} \{b'_e \mid b' \in B \wedge b_s + T \in [b'_s, b'_e)\} \\ \cup \max\{b'_e \mid b' \in B\} \end{pmatrix}).$$

[1] https://twitter.com

MediaEval 2015 Workshop, Sept. 14-15, 2015, Wurzen, Germany

3. EXPERIMENTS

3.1 Dataset

The dataset used in the SAVA task is a subset of collection of 4021 hours of video broadcasted by the BBC [2]. This subset consists of a dev set (37 videos, 37 hours) and a test set (33 videos, 31 hours). The experiments presented here make use of manually transcribed subtitles provided by the BBC and use shot boundaries that ship with the SAVA dataset.

3.2 Setup

In this paper, we have tested two weighing functions, w_1 and w_2, each being submitted as a separate run in the benchmark. The first function takes the popularity of each keyphrase in a shot into account, while the second function only considers the number of different keyphrases. They are defined as follows:

$$w_1(k) = \begin{cases} (\ln \circ q)(k) & \text{if } q(k) > 0 \\ 0 & \text{otherwise} \end{cases}$$

$$w_2(k) = \begin{cases} 1 & \text{if } q(k) > 0 \\ 0 & \text{otherwise} \end{cases}$$

Furthermore, the keyphrase extraction used in our method only considered words of length 2 to 40 characters and ignored stopwords (using `nltk`'s default stopwords list) and words that were written in all capital letters. The latter filter was used to ignore words appearing in descriptive subtitles for the hearing impaired, such as "APPLAUSE". This resulted in 37,154 nounphrase candidates for both dev and test set. To reduce the number of queries to be crawled, we pruned the list of keyphrases using the following heuristics:

- The phrase should contain at least one capital letter;
- The phrase may not start or end with a stopword;
- The phrase does not start with a quote;
- The phrase does not contain periods or commas.

The last two heuristics were put in place to deal with tokenization mistakes. Applying these pruning heuristics, we reduced the number of phrases from 37,154 to 5,663.

Querying Twitter for these 5,663 phrases for the period between 2008-04-01 to 2008-07-31 (corresponding to the original broadcast dates of the dataset) resulted in 66,934 tweets. We did not impose any language or geographic restrictions and issued queries as specified in Section 2. In this querying process we used a cut-off point to speed up the crawling process. After collecting more than 150 tweets for a query, the process was stopped. This means that in this experiment $0 \leq q(k) \leq 150$. The software we used to unrestrictedly crawl Twitter is available on GitHub.[2]

3.3 Results

Submitted anchors from all submissions task were pooled and assessed by crowdsourcing workers on Amazon Mechanical Turk [2]. We note that our method itself does not take overlap of anchor segments into account, but that this was addressed by the automatic evaluation of our submission. The results of the two runs are summarized in Table 1, showing precision at 10 (P@10), recall and mean reciprocal rank (MRR) averaged over the 33 videos in the test set.

Table 1: Run results averaged over 33 videos in the test set

	Precision	Recall	MRR	Unjudged	
	@10			@10	@1000
Run w_1	0.55758	0.47496	0.87879	1.21212	6.75758
Run w_2	0.50000	0.43224	0.93939	1.39394	8.39394

From the results, we can see that run 1, which uses the w_1 weighting function that assigns a higher weight to more popular keyphrases, appears to have a better precision and recall than run 2, which uses the w_2 weighing function that treats all keyphrases equally. Run 2 on the other hand achieves a higher MRR. However, in both runs, the MRR never drops below 0.5, indicating that the first relevant anchor is always amongst the first two results.

3.4 Analysis

It is of most interest to see when our proposed method performs best and when it does not. For this analysis we will look at the correlation between different performance metrics and video features.

Our first observation is that P@10 and recall appear to be positively correlated, with Pearson's r being 0.42 (with $p < 0.02$) and 0.76 (with $p \ll 0.01$) for run 1 and run 2, respectively. This suggests that some videos are easier and others are harder for our method to get right. We found that video length correlate positively with P@10 (0.56 and 0.54 for the respective runs) and that this correlation is significant ($p < 0.01$, for both runs). No conclusions could be drawn for the correlation between video length and recall, however. The answer to why our method seem to be in favor of longer videos could be found in the following two correlations. First, our method tends to perform better (P@10) when the unpruned list of different extracted keyphrases is longer (correlation of 0.58 ($p < 0.001$) and 0.51 ($p < 0.01$) for run 1 and 2, respectively). Second, we can extract almost undoubtedly more keyphrases from longer videos, as the correlation between these two features is significant (0.83 with $p \ll 0.001$ for both runs). Our final observation is that, interestingly, P@10 also seems to correlate with to something related to how we score individual segments, namely the sum of the summed keyphrases weights ($W(b)$) of the first 10 results: 0.44 ($p < 0.02$) and 0.48 ($p < 0.01$) for run 1 and 2, respectively.

4. CONCLUSION

We have explored a method for predicting anchors in television broadcasts by measuring interrogative activity on microblogs. Using the simplifying assumption that a shot is important if its subtitles contain keyphrases that appear in questions asked on a microblog platform, our method is able to achieve promising performance. Suggestions for future work include the following. We believe that finetuning the keyphrase extraction process and looking at incorporating tf-idf could help with dealing with generic keyphrases that sometimes are extracted (e.g., "Good evening"). Furthermore, investigating the impact of narrowing and broadening a query's date range seems interesting to see what type questions are important and how to strike the balance between questions which are ephemeral and questions which are evergreen. Last but not least, we could look into personalization of the method, such as localizing the query to a geographic region.

[2] https://github.com/ShinNoNoir/twitterwebsearch

5. REFERENCES

[1] S. Bird, E. Loper, and E. Klein. *Natural language processing with Python*. O'Reilly Media, Inc., 2009.

[2] M. Eskevich, R. Aly, R. Ordelman, D. N. Racca, S. Chen, and G. J. F. Jones. SAVA at MediaEval 2015: Search and anchoring in video archives. In *MediaEval*, Wurzen, Germany, September 2015.

[3] S. N. Kim, T. Baldwin, and M.-Y. Kan. Evaluating n-gram based evaluation metrics for automatic keyphrase extraction. In *Proceedings of the 23rd International Conference on Computational Linguistics*, COLING '10, pages 572–580, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics.