7-2022

Towards aligning slides and video snippets: Mitigating sequence and content mismatches

Ziyuan LIU
Singapore Management University, ziyuan.liu.2018@scis.smu.edu.sg

Hady W. LAUW
Singapore Management University, hadywlauw@smu.edu.sg

Citation

LIU, Ziyuan and LAUW, Hady W.. Towards aligning slides and video snippets: Mitigating sequence and content mismatches. (2022). *23rd International Conference on Artificial Intelligence in Education, AIED 2022: Durham, July 27-31: Proceedings*. 13355, 533-539.
Available at: https://ink.library.smu.edu.sg/sis_research/7600

Towards Aligning Slides and Video Snippets: Mitigating Sequence and Content Mismatches

Ziyuan Liu(✉) and Hady W. Lauw(✉)

School of Computing and Information Systems, Singapore Management University,
Singapore, Singapore
{ziyuan.liu.2018,hadywlauw}@smu.edu.sg

Abstract. Slides are important form of teaching materials used in various courses at academic institutions. Due to their compactness, slides on their own may not stand as complete reference materials. To aid students' understanding, it would be useful to supplement slides with other materials such as online videos. Given a deck of slides and a related video, we seek to align each slide in the deck to a relevant video snippet, if any. While this problem could be formulated as aligning two time series (each involving a sequence of text contents), we anticipate challenges in generating matches arising from differences in content coverage and sequence of content between slide deck-video pairs. To mitigate these challenges, we propose a two-stage algorithm that builds on time series alignment to filter out irrelevant content and to align out-of-sequence slide deck and video pairs. We experiment with real-world datasets from openly available lectures, which have been manually annotated with start and end times of each slide in the videos to facilitate the evaluation of matches.

Keywords: Slide to video alignment · Dynamic time warping · Sequence mismatch · Content mismatch

1 Introduction

Many instructors use slides as teaching aid, and often make these available to students as reference material. The compact and terse nature may render slides, on their own, inadequate for the latter function of reference materials. Students may need to rely on additional outside materials, such as videos that can be found in course webpages, massive open online courses, or video sites.

Some works attempt to augment academic or educational materials with additional content [3,8]. Adamson et al. [1] set out a means of automatically generating questions to support instruction and learning, while others seek to support teaching by generating answers to questions [2,13].

We envision a system where a student who is reviewing a deck of slides can be pointed to a snippet of a video that is relevant to the slide currently being viewed. Given a video relevant to a deck of lecture slides, we seek to align each

M. M. Rodrigo et al. (Eds.): AIED 2022, LNCS 13355, pp. 533–539, 2022.
https://doi.org/10.1007/978-3-031-11644-5_45

slide to a snippet within the video. This also involves detecting when there is no snippet within the video relevant to the slide. The technical challenge here concerns aligning two collections of different modalities, e.g., slides and videos. Our approach treats both slides and videos as time series of text contents.

As the first contribution in this paper, we propose SEQUENTIALIGN (see Sect. 2), a methodology for aligning a slide deck and a video that mitigates sequence mismatch and content mismatch. As a second contribution, we build an annotated dataset of aligning slides to video snippets from pairings of educational slides and videos from computer science topics such as artificial intelligence, operating systems, and systems programming. As a third contribution, we empirically validate the approach (see Sect. 3) on the afore-mentioned annotated data against comparable baselines.

2 Methodology: Sequentialign

Data. Our data consists of slide deck-video pairs, each consisting of a slide deck and a related video. Within each pair, at least one slide in the deck matches one snippet in the video. We work with the textual contents of the slides and the videos (i.e., transcripts). Each slide deck $\mathbf{s}$ consists of a number of slides, given by a sequence of vectors $\mathbf{s} = \{\boldsymbol{s}_1, \boldsymbol{s}_2, \boldsymbol{s}_3 ... \boldsymbol{s}_m\}$, each representing the textual context of a single slide. These vectors could be based on bag-of-words representation such as tf-idf or word embeddings [9]. In turn, each video $\mathbf{v}$ is divided into video snippets of a specified equal duration,[1] given by a sequence of vectors $\mathbf{v} = \{\boldsymbol{v}_1, \boldsymbol{v}_2, \boldsymbol{v}_3 ... \boldsymbol{v}_n\}$ each representing the transcribed content of a single snippet.

Problem. For each slide deck-video pair, our task is to find a set of matches (an alignment) between slides and video snippets, such as $(s_i \rightarrow [v_{j1}, v_{j2} ... v_{jn}])$, where $[v_{j1}, v_{j2} ... v_{jn}]$ is a set of video snippets matched to s_i. Each slide may be assigned to 0 or more snippets. Each snippet can be assigned to 0 or 1 slide.

Dynamic Time Warping. We can view a deck of slides as a time series whereby each slide is a time point. Similarly, each snippet is a time point within a video time series. Among techniques for measuring the similarity between two time series [10], Dynamic Time Warping (DTW) is known as a robust way to measure similarity between two time series that vary in speed [6]. It also produces an alignment of time points between the two respective time series. Without losing generality, we build our proposed algorithm using DTW as a building block. However, DTW has a couple of constraints that render it unsuitable for direct use. The monotonicity property requires that indices of successive matches on either sequence should be monotonically increasing, thereby forcing false matches when the two time series are out of sequence. The continuity property requires matched indices on each sequence to increase one at a time, thereby continuing

[1] In our experiments, each basic unit of video snippet is of 30-s duration. The last snippet in a video may be shorter.

false matches over periods of irrelevant snippets. To counter these, we propose SequentiAlign that addresses the sequence and content mismatches.

2.1 Mitigating Sequence Mismatch

Consider a slide deck-video pair which cover a similar set of topics. If sequence were not informative, we may consider matching using a distance measure alone. A naive means of doing this would be to divide the slides or the videos into blocks, and perform minimum weight bipartite matching [4]. Between each of the blocks, we calculate a distance and find a matching that minimizes the total distance.

While the overall sequence of a slide deck and video might be different, there may be common local subsequences in which the flow of topics follow a similar, logical order. It may be useful to use DTW as a distance measure locally, while relying on bipartite matching globally. This forms the basis for the alignment subroutine of SequentiAlign. Our alignment subroutine divides both the slides and the video snippets into a number of blocks set in a 2-dimensional grid (given by the grid factor, g), each containing an equal number of slides or snippets, as the case may be. For every cell in the grid, each representing a possible match between a slide block and a video snippet block, we run DTW locally, giving each cell a warping distance. The Hungarian algorithm [7] is used to find a minimum-weight bipartite matching between the 2 axes on the grid, to identify a set of cells representing one-to-one matches with the lowest total distance measure.

While the initial grids enforce equal-sized blocks, to model more natural alignment that may involve different-sized blocks, after the alignment subroutine obtains the matches given by the bipartite matching, it runs the DTW algorithm again on adjacent slide blocks in the matches, and the warping path returned is used to adjust the boundaries between their matched video snippet blocks, while leaving the slide block boundaries unchanged. The intuition is that this process will break through the rigidity of the uniform length of blocks, and allow snippets on the cell boundaries to be assigned to the correct slides, while the bipartite matching between slide blocks and video blocks makes it possible for common subsequences to be matched together out-of-sequence between a slide deck-video pair, even as the monotonicity constraint of DTW is respected locally.

What remains is the determination of the value of g, which we consider a hyperparameter. Our approach is to search from 1 to two-thirds of number of slides, and pick the value of g which yields the minimum distance measure.

2.2 Mitigating Content Mismatch

To mitigate the content mismatch, we identify irrelevant slides and video snippets and remove them before alignment. For one naive approach to identify an irrelevant slide, we can consider its minimum distance to any snippet and impose a maximum threshold. For another, we can let each video identify its closest slide, and remove any slide not picked by any video. Analogously, we can attempt to identify an irrelevant video snippet. Both look at each slide (resp. snippet) independently of any other slide in the deck (resp. snippet in the video).

We postulate that to identify whether a slide is relevant, we need to consider the neighbouring slides, and whether as a group of slides they may match a sequence of video snippets as well. To allow the consideration of multiple target window sizes, we introduce the concept of *relevance score.* For video snippets, the primary component of this score is the number of best-match windows it is part of. For slides, the primary component of this score is the number of video snippets it is matched to across all queries. The raw scores are adjusted for the distance between each query and its best-match window, and the distance between each video snippet and query slide pair within best-match windows.

Drawing from [12], our subsequence search subroutine uses DTW as a subroutine. For each slide, it constructs a 'query' by taking the slide itself and a number of subsequent slides, given by the length of the query, r. DTW is run between the query and target windows of video snippets of length q, with starting index incremented by 1 for each successive window. For each query, we match the first slide in the query with the sequence of snippets starting with the first snippet of the best-match window, and the snippet immediately before that identified by the path as the starting point of the second slide in the query in the window. If the first snippet matched to second slide is the same as that of the first, we do not match the first slide with any snippet, and return an empty set.

To filter out irrelevant content, we run the subsequence search subroutine multiple times, using varying window sizes for the target windows of video snippets. For each query q, we take note of the best match window, the video snippets matched to each slide in the query $[v_1, v_2 ... v_n]$, the total distance (cost) between the query and best match window, and the cosine distances (distance) between each slide and the video snippets it is matched to. Having calculated relevance scores for all slides and video segments, we set a percentile threshold for determining relevance, and remove slides and video segments with a relevance score below the relevance score value at the percentile threshold. For instance, if the 25th percentile of slide relevance scores is 75.5, slides with relevance scores below 75.5 are labelled as irrelevant and removed. We name filtering subroutines (and the SEQUENTIALIGN implementation it is used in) according to the percentile threshold of relevance scores used to identify irrelevant slides. For example, SEQUENTIALIGN-33 combines the filtering subroutine with 33rd percentile threshold with the alignment algorithm described in the previous section. The use of percentile threshold, instead of absolute threshold, is to guard against different levels of text similarities across domains.

3 Experiments

Our objective is to evaluate efficacy of various methods at producing alignments between video and slides on real-world datasets.

Datasets. We annotate 6 datasets containing slide-deck video pairs from publicly available lectures, as summarized in Table 1, covering subjects such as Artificial Intelligence, Operating Systems, and Systems Programming in C/C++.

Each slide is labelled with start and end times corresponding to the video portion which in the opinion of the annotator best matches the slide content. The datasets have content and sequence mismatch between slide decks and videos.

Baselines. We compare our SequentiAlign algorithm with these baselines:

Table 1. Summary of datasets

	Video course	Slides course	Pairs	Slide count		Video duration (s)	
				Mean	Median	Mean	Median
BerkeleyStanford-AI	Berkeley CS188	Stanford CS221	8	39.5	36.5	4735.8	4863.0
StanfordBerkeley-AI	Stanford CS221	Berkeley CS188	8	42.3	42.0	4149.6	4140.0
BerkeleyVirginia-OS	Berkeley CS162	UVirginia CS4414	8	90.5	87.0	5203.3	5233.5
VirginiaBerkeley-OS	UVirginia CS4414	Berkeley CS162	8	66.1	61.0	4551.9	4489.0
CMUCornell-C	CMU 15213	Cornell CS4414	5	44.8	49.0	3305.6	2980.0
CornellCMU-C	Cornell CS4414	CMU 15213	5	56.4	56.0	4295.0	4613.0

Table 2. Performance on various slide deck-video pairs from different sources

	Artificial intelligence				Operating systems				Systems programming in C			
	BerkeleyStanford		StanfordBerkeley		BerkeleyVirginia		VirginiaBerkeley		CMUCornell		CornellCMU	
	Acc	IoU	Acc	IoU	Acc	IoU	Acc	IoU	Acc	IoU	Acc	IoU
Random	0.004	0.001	0.005	0.003	0.006	0.004	0.017	0.004	0.001	0.002	0.004	0.001
DTW	0.027	0.011	0.051	0.022	0.010	0.007	0.048	0.011	0.087	0.040	0.137	0.061
HMM+IBM1	0.003	0.001	0.017	0.007	0.010	0.007	0.012	0.002	0.030	0.016	0.008	0.003
SequentiAlign-25	0.109	0.126	0.189	0.172	0.234	0.168	0.236	0.167	0.125	0.211	0.179	0.198
SequentiAlign-33	0.211	0.190	0.253	0.224	0.305	0.230	0.300	0.230	0.160	0.260	0.241	0.276
SequentiAlign-50	0.407	0.314	0.389	0.343	0.444	0.365	0.406	0.365	0.414	0.331	0.366	0.405

HMM+IBM1. The closest related work in terms of task is the HMM+IBM1 [11]. We align video snippets with slides using a window of jump probabilities $[-2, 2]$. It mainly targets sequence alignment without targeting content mismatch.

Dynamic Time Warping (DTW). To evaluate the performance without the mitigation of the sequence and content mismatch provided by SequentiAlign over the base alignment algorithm, we compare to the vanilla DTW.

Random. We split the video snippets into as many segments as there are slides, and assign each segment randomly to a slide.

Metrics. We use the following metrics that are commonly associated with multimedia retrieval or alignment:

Accuracy (Acc). Accuracy is the number of seconds in the video with true positive and true negative alignment outcomes, over the duration of the video in seconds. True positive is defined as seconds correctly aligned to the right slide. True negative is defined as seconds which are irrelevant to any slide and correctly identified. We average accuracy across all slide deck-video pairs.

Intersection over Union (IoU). Following [5], for each slide, we measure the intersecting duration between the predicted and the ground truth video spans and divide this by the union. For true negative hits, the IoU value is taken to be 1. We then average this IoU over the slides in a deck, and over the decks.

Empirical Results. In Sect. 2.2, we describe dealing with content mismatch by filtering out irrelevant content that involves specifying a percentile threshold, yielding the various SequentiAlign variants (at 25th, 33rd, and 50th percentiles). The results are shown in Table 2. The SequentiAlign variants tend to outperform over the baselines across all the datasets here. DTW and HMM+IBM1 perform rather poorly due to the considerable content and sequence mismatch in these datasets. The performance of SequentiAlign steadily improves as we remove more irrelevant content.

4 Conclusion

In conclusion, we have proposed a framework for the generation of matches between slide deck-video pairs. To mitigate the content mismatch and sequence mismatch problems which can cause an unmodified DTW algorithm to be less suitable for the task of generating matches, we propose a 2-step solution, by first identifying probable irrelevant slides using a subsequence search approach, and then focusing on finding good matches despite the sequence mismatch problem, using the alignment subroutine. Experiments on slides and videos from real courses show promise. We identify several directions for future work. In our experiments, we produce alignments for slide decks with a single video. We could run SequentiAlign across several videos to find more matches for a given slide. Being more aggressive with content filtering may achieve higher quality matches with smaller quantity from each video but higher quantity across videos.

Acknowledgements. This research is supported by the Centre for Teaching Excellence at Singapore Management University under its Educational Research Fellowship Programme.

References

1. Adamson, D., Bhartiya, D., Gujral, B., Kedia, R., Singh, A., Rosé, C.P.: Automatically generating discussion questions. In: Lane, H.C., Yacef, K., Mostow, J., Pavlik, P. (eds.) AIED 2013. LNCS (LNAI), vol. 7926, pp. 81–90. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-39112-5_9
2. Atapattu, T., Falkner, K., Falkner, N.: Educational question answering motivated by question-specific concept maps. In: Conati, C., Heffernan, N., Mitrovic, A., Verdejo, M.F. (eds.) AIED 2015. LNCS (LNAI), vol. 9112, pp. 13–22. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-19773-9_2
3. Csomai, A., Mihalcea, R.: Linking educational materials to encyclopedic knowledge. Front. Artif. Intell. Appl. **158**, 557 (2007)

4. Duan, R., Pettie, S.: Linear-time approximation for maximum weight matching. J. ACM (JACM) **61**(1), 1–23 (2014)
5. Gao, J., Sun, C., Yang, Z., Nevatia, R.: TALL: temporal activity localization via language query. In: ICCV, pp. 5267–5275 (2017)
6. Keogh, E.J., Pazzani, M.J.: Scaling up dynamic time warping for datamining applications. In: Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 285–289 (2000)
7. Kuhn, H.W., Yaw, B.: The Hungarian method for the assignment problem. Nav. Res. Logist. Q. **2**, 83–97 (1955)
8. Labhishetty, S., Bhavya, Pei, K., Boughoula, A., Zhai, C.: Web of slides: automatic linking of lecture slides to facilitate navigation. In: ACM L@S (2019)
9. Le, Q., Mikolov, T.: Distributed representations of sentences and documents. In: International Conference on Machine Learning, pp. 1188–1196. PMLR (2014)
10. Morse, M.D., Patel, J.M.: An efficient and accurate method for evaluating time series similarity. In: SIGMOD, pp. 569–580 (2007)
11. Naim, I., Song, Y.C., Liu, Q., Kautz, H., Luo, J., Gildea, D.: Unsupervised alignment of natural language instructions with video segments. In: AAAI (2014)
12. Rakthanmanon, T., et al.: Searching and mining trillions of time series subsequences under dynamic time warping. In: KDD (2012)
13. Zylich, B., Viola, A., Toggerson, B., Al-Hariri, L., Lan, A.: Exploring automated question answering methods for teaching assistance. In: Bittencourt, I.I., Cukurova, M., Muldner, K., Luckin, R., Millán, E. (eds.) AIED 2020. LNCS (LNAI), vol. 12163, pp. 610–622. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-52237-7_49