A Neural Multi-sequence Alignment TeCHnique (NeuMATCH)

Pelin Dogan[1,4*] Boyang Li[2] Leonid Sigal[3] Markus Gross[1,4]
[1]ETH Zürich [2]Liulishuo AI Lab [3]University of British Columbia [4]Disney Research
{pelin.dogan, grossm}@inf.ethz.ch, albert.li@liulishuo.com, lsigal@cs.ubc.ca

Abstract

The alignment of heterogeneous sequential data (video to text) is an important and challenging problem. Standard techniques for this task, including Dynamic Time Warping (DTW) and Conditional Random Fields (CRFs), suffer from inherent drawbacks. Mainly, the Markov assumption implies that, given the immediate past, future alignment decisions are independent of further history. The separation between similarity computation and alignment decision also prevents end-to-end training. In this paper, we propose an end-to-end neural architecture where alignment actions are implemented as moving data between stacks of Long Short-term Memory (LSTM) blocks. This flexible architecture supports a large variety of alignment tasks, including one-to-one, one-to-many, skipping unmatched elements, and (with extensions) non-monotonic alignment. Extensive experiments on semi-synthetic and real datasets show that our algorithm outperforms state-of-the-art baselines.

1. Introduction

Sequence alignment (see Figure 1) is a prevalent problem that finds diverse applications in molecular biology [27], natural language processing [3], historic linguistics [33], and computer vision [7]. In this paper, we focus on aligning heterogeneous sequences with complex correspondences. Heterogeneity refers to the lack of an obvious surface matching (a literal similarity metric between elements of the sequences). A prime example is the alignment between visual and textual content. Such alignment requires sophisticated extraction of comparable feature representations in each modality, often performed by a deep neural network.

A common solution to the alignment problem consists of two stages that are performed separately: (1) the learning of a similarity metric between elements in the sequences and (2) finding the optimal alignment between the sequences. Alignment techniques based on dynamic programming, such as Dynamic Time Warping (DTW) [4] and Canonical Time Warping (CTW) [59], are widely popular. In a simple form, DTW can be understood as finding the shortest path where the edge costs are computed with the similarity metric, so the decision is Markov. Variations of DTW [43, 60] accommodate some degrees of non-monotonicity (see Figure 1 (c)). In all cases, these approaches are disadvantaged by the separation of the two stages. Conceptually, learning a metric that directly helps to optimize alignment should be beneficial. Further, methods with first-order Markov assumptions take only limited local context into account, but contextual information conducive to alignment may be scattered over the entire sequence. For example, knowledge of the narrative structure of a movie may help to align shots to their sentence descriptions.

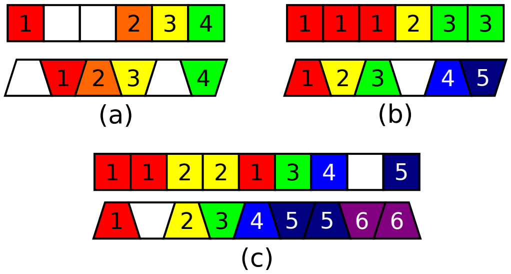


Figure 1: Types of sequence correspondence. Matching blocks in two sequences have identical colors and numbers. (a) A one-to-one matching where the white blocks do not match anything. (b) A one-to-many matching where one block on the bottom sequence matches multiple blocks on the top. (c) A non-monotonic situation where the matching does not always proceed strictly from left to right due to the red-1 block after the yellow-2 on top.

To address these limitations, we propose an end-to-end differentiable neural architecture for heterogeneous sequence alignment, which we call NeuMATCH. The NeuMATCH architecture represents the current state of the workspace using four Long Short-term Memory (LSTM) chains: two for the partially aligned sequences, one for the matched content, and one for historical alignment decisions.

[*] The technique was conceived when all authors worked for Disney Research.

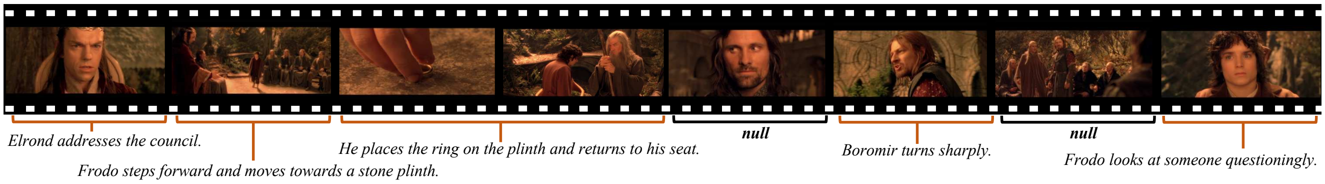

Elrond addresses the council.
Frodo steps forward and moves towards a stone plinth.
He places the ring on the plinth and returns to his seat.
null
Boromir turns sharply.
null
Frodo looks at someone questioningly.

Figure 2: An example alignment between clip sequence and text sequence (from the dataset HM-2 in Section 4.1).

The four recurrent LSTM networks collectively capture the decision context, which is then classified into one of the available alignment actions. Compared to the traditional two-stage solution, the network can be optimized end-to-end. In addition, the previously matched content and the decision history inform future alignment decisions in a non-Markov manner. For example, if we match a person's face with the name Frodo at the beginning of a movie, we should be able to identify the same person again later (Figure 2). Alternatively, if the input sequences are sampled at different rates (e.g., every third video clip is matched to text), the decision history can help to discover and exploit such regularities.

Although the proposed framework can be applied to different types of sequential data, in this paper, we focus on the alignment of video and textual sequences, especially those containing narrative content like movies. This task is an important link in joint understanding of multimodal content [16] and is closely related to activity recognition [10, 51], dense caption generation [25], and multimedia content retrieval [22, 46]. The reason for choosing narrative content is that it is among the most challenging for computational understanding due to a multitude of causal and temporal interactions between events [38]. Disambiguation is difficult with needed contextual information positioned far apart. Thus, narrative contents make an ideal application and testbed for alignment algorithms.

Contributions. The contributions of this paper are two-fold. First, we propose a novel end-to-end neural framework for heterogeneous multi-sequence alignment. Unlike prior methods, our architecture is able to take into account rich context when making alignment decisions. Extensive experiments illustrate that the framework *significantly* outperforms traditional baselines in accuracy. Second, we annotate a new dataset[1] containing movie summary videos and share it with the research community.

2. Related Work

Our goal of video-text alignment is related to multiple topics. We briefly review the most relevant literature below.

Unimodal Representations. It has been observed that deep convolutional neural networks (CNNs), such as VGG [39], ResNet [18], GoogLeNet [41], and even automatically learned architectures [61], can learn image features that are transferable to many different vision tasks [13, 57]. Generic representations for video and text have received comparatively less attention. Common encoding techniques for video include pooling [48] and attention [54, 56] over frame features, neural recurrence between frames [12, 34, 47], and spatiotemporal 3D convolution [45]. On the language side, distributed word representations [30, 32] are often used in recurrent architectures in order to model sentential semantics. When coupled with carefully designed training objectives, such as Skip-Thought [23] or textual entailment [6, 8], they yield effective representations that generalize well to other tasks.

Joint Reasoning of Video and Text. Popular research topics in joint reasoning and understanding of visual and textual information include image captioning [21, 29, 50, 54], retrieval of visual content [26], and visual question answering [2, 36, 53]. Most approaches along these lines can be classified as belonging to either (i) joint language-visual embeddings or (ii) encoder-decoder architectures. The joint *vision-language embeddings* facilitate image/video or caption/sentence retrieval by learning to embed images/videos and sentences into the same space [31, 44, 52, 55]. For example, [19] uses simple kernel CCA and in [17] both images and sentences are mapped into a common semantic *meaning* space defined by object-action-scene triplets. More recent methods directly minimize a pairwise ranking function between positive image-caption pairs and contrastive (non-descriptive) negative pairs; various ranking objective functions have been proposed including max-margin [22] and order-preserving losses [46]. The *encoder-decoder* architectures [44] are similar, but instead attempt to encode images into the embedding space from which a sentence can be decoded. Applications of these approaches for video captioning and dense video captioning (multiple sentences) were explored in [31] and [58] respectively, for video retrieval in [12], and for visual question answering in [1]. In this work, we jointly encode the video and textual input as part of the decision context. Instead of decoding alignment decisions one by one with RNNs, we gather the most relevant contexts for every alignment decision and directly predict the decision from those.

Video-text alignment. Under the dynamic time warping framework, early works on video/image-text alignment adopted a feature-rich approach, utilizing features from di-

[1] https://github.com/pelindogan/NeuMATCH

	[37]	[60]	[43]	[42]	[5]	**NeuMATCH**
Method	DTW	CRF Chain	DP	DP	QIP	Neural
End-to-end Training	No	No	No	No	No	Yes
Historic Context	Markov	Markov + Convolution on Similarity	Markov	Markov	global	high order
Supports Non-monotonicity	No	Yes	Yes	No	No	Yes*
Visual/Textual Granularity	fine	medium	coarse	fine	fine	fine

Table 1: Comparison of existing video-text alignment approaches. Prior method are based on DTW/Dynamic Programming (DP), Conditional Random Field (CRF) and Convex Quadratic Programming (CQP). *Non-monotonicity requires extensions in Appendix A.

alogs and subtitles [9, 15, 42], location, face and speech recognition [37], as well as nouns and pronouns between text and objects in the scenes [11, 24, 26, 28].

Tapaswi *et al.* [42] present an approach to align plot synopses with the corresponding shots with the guidance of subtitles and facial features from characters. They extend the DTW algorithm to allow one-to-many matching. In [43], Tapaswi *et al.* present another extension to allow non-monotonic matching in the alignment of book chapters and video scenes. The above formulations make use of the Markov property, which enables efficient solutions with dynamic programming (DP). At the same time, the historic context being considered is limited. [60] develops neural approach for the computation of similarities between videos and book chapters, using Skip-Thought vectors [23]. In order to capture historic context, they use a convolutional network over a similarity tensor. The alignment is formulated as a linear-chain Conditional Random Field (CRF), which again yields efficient solution from DP. Although this method considers historic context, the alignment and similarity are still computed separately.

Bojanowski *et al.* [5] formulate alignment as quadratic integer programming (QIP) and solve the relaxed problem. Weak supervision can be introduced as optimization constraints. This method considers the global context, but relates the video and text features by a linear transformation and does not consider non-monotonic alignment. Table 1 compares key aspects of these methods.

In summary, existing approaches perform the alignment in two separate stages: (1) extracting visual and textual features in such a way as to have a well defined metric, and (2) performing the alignment using this similarity (and possibly additional side information). We propose an end-to-end differentiable neural architecture that considers more than the local similarities. Inspired by LSTM-powered shift-reduce language parsers [14, 20], we augment LSTM networks with stack operations, such as pop and push. The advantage of this setup is that the most relevant video clips, sentences, and historic records are always positioned closest to the prediction.

3. Approach

We now present NeuMATCH, a neural architecture for temporal alignment of heterogeneous sequences. While the network is general, for this paper we focus specifically on the video and textual sequence alignment. The video sequence consists of a number of consecutive video clips $\mathcal{V} = \{V_i\}_{i=1\ldots N}$. The textual sequence consists a number of consecutive sentences $\mathcal{S} = \{S_i\}_{i=1\ldots M}$. Our task is to align these two sequences by, for example, finding a function π that maps an index of the video segment to the corresponding sentence: $\langle V_i, S_{\pi(i)} \rangle$. An example input for our algorithm can be a movie segmented into individual shots and the accompanying movie script describing the scenes and actions, which are broken down into sentences (Figure 2). The video segmentation could be achieved using any shot boundary detection algorithm; NeuMATCH can handle one-to-many matching caused by over-segmentation.

We observe that the most difficult sequence alignment problems exhibit the following characteristics. First, heterogeneous surface forms, such as video and text, can conceal the true similarity structure, which suggests a satisfactory understanding of the entire content may be necessary for alignment. Second, difficult problems contain complex correspondence like many-to-one matching and unmatched content, which the framework should accommodate. Third, contextual information that are needed for learning the similarity metric are scattered over the entire sequence. Thus, it is important to consider the history and the future when making the alignment decision and to create an end-to-end network where gradient from alignment decisions can inform content understanding and similarity metric learning.

The NeuMATCH framework copes with these challenges by explicitly representing the state of the entire workspace, including the partially matched input sequences and historic alignment decisions. The representation employs four LSTM recurrent networks, including the input video sequence (Video Stack), the input textual sequence (Text Stack), previous alignment actions (Action Stack) as well as previous alignments themselves (Matched Stack). Figure 3 shows the NeuMATCH architecture.

We learn a function that maps the state of workspace Ψ_t to an alignment action A_t at every time step t. The action

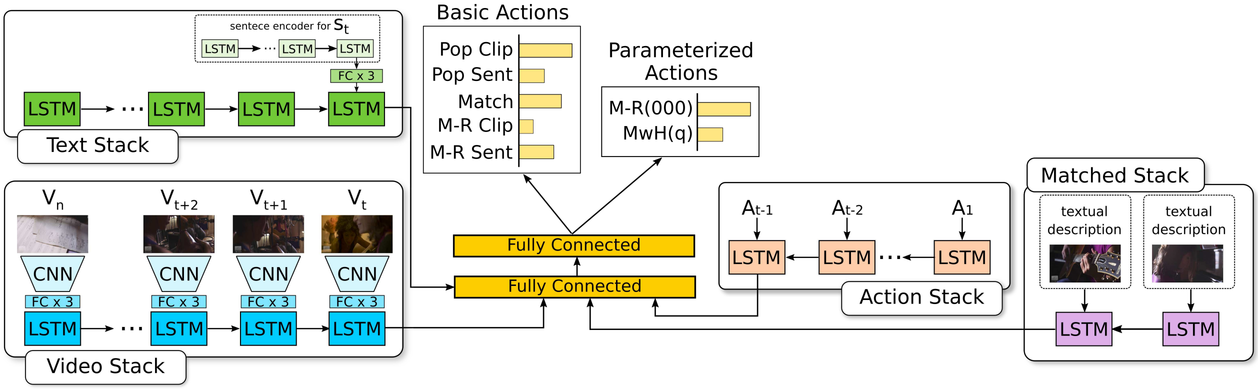


Figure 3: The proposed NeuMATCH neural architecture. The current state as described by the four LSTM chains is classified into one of the alignment decisions. Parameterized actions are explained and illustrated in Appendix A and Table 7.

A_t manipulates the content of the LSTM networks, resulting in a new state Ψ_{t+1}. Executing a complete sequence of actions produces an alignment of the input. The reader may recognize the similarity with policy gradient methods [40]. As the correct action sequence is unique in most cases and can be easily inferred from the ground-truth labels, in this paper, we adopt a supervised learning approach.

The alignment actions may be seen as stack operations because they either remove or insert an element at the first position of the LSTM network (except for non-monotonic matching discussed in Appendix A). For example, elements at the first position can be removed (*popped*) or *matched*. When two elements are matched, they are removed from the input stacks and stored in the Matched Stack.

3.1. Language and Visual Encoders

We first create encoders for each video clip and each sentence. After that, we perform an optional pre-training step to jointly embed the encoded video clips and sentences into the same space. While the pre-training step produces a good initialization, the entire framework is trained end-to-end, which allows the similarity metric to be specifically optimized for the alignment task.

Video Encoder. We extract features using the activation of the first fully connected layer in the VGG-16 network [39], which produces a 4096-dim vector per frame. As each clip is relatively short and homogeneous, we perform mean pooling over all frames in the video, yielding a feature vector for the entire clip. This vector is transformed with three fully connected layers using the ReLU activation function, resulting in encoded video vector v_i for the i^{th} clip.

Sentence Encoder. The input text is parsed into sentences $S_1 \dots S_M$, each of which contains a sequence of words. We transform each unique word into an embedding vector pre-trained using GloVe [32]. The entire sentence is then encoded using a 2-layer LSTM recurrent network, where the hidden state of the first layer, $h_t^{(1)}$, is fed to the second layer:

$$h_t^{(1)}, c_t^{(1)} = \text{LSTM}(x_t, h_{t-1}^{(1)}, c_{t-1}^{(1)}) \tag{1a}$$

$$h_t^{(2)}, c_t^{(2)} = \text{LSTM}(h_t^{(1)}, h_{t-1}^{(2)}, c_{t-1}^{(2)}) \ , \tag{1b}$$

where $c_t^{(1)}$ and $c_t^{(2)}$ are the memory cells for the two layers, respectively; x_t is the word embedding for time step t. The sentence is represented as the vector obtained by the transformation of the last hidden state $h_t^{(2)}$ by three fully connected layers using ReLU activation function.

Encoding Alignment and Pre-training. Due to the complexity of the video and textual encoders, we opt for pre-training that produces a good initialization for subsequent end-to-end training. For a ground-truth pair (V_i, S_i), we adopt an asymmetric similarity proposed by [46]

$$F(v_i, s_i) = -||\max(0, v_i - s_i)||^2 \ . \tag{2}$$

This similarity function takes the maximum value 0, when s_i is positioned to the upper right of v_i in the vector space. That is, $\forall j, s_{i,j} \geq v_{i,j}$. When that condition is not satisfied, the similarity decreases. In [46], this relative spatial position defines an entailment relation where v_i entails s_i. Here the intuition is that the video typically contains more information than being described in the text, so we may consider the text as entailed by the video.

We adopt the following ranking loss objective by randomly sampling a contrastive video clip V' and a contrastive sentence S' for every ground truth pair. Minimizing the loss function maintains that the similarity of the contrastive pair is below true pair by at least the margin α.

$$\begin{aligned}\mathcal{L} = \sum_i \big(& \mathbb{E}_{v' \neq v_i} \max\{0, \alpha - F(v_i, s_i) + F(v', s_i)\} \\ & + \mathbb{E}_{s' \neq s_i} \max\{0, \alpha - F(v_i, s_i) + F(v_i, s')\}\big)\end{aligned} \tag{3}$$

Note the expectations are approximated by sampling.

3.2. The NeuMATCH Alignment Network

With the similarity metric between video and text acquired by pre-training, a naive approach for alignment is to maximize the collective similarity over the matched video clips and sentences. However, doing so ignores the temporal structures of the two sequences and can lead to degraded performance. NeuMATCH considers the history and the future by encoding input sequences and decision history with LSTM networks.

LSTM Stacks. At time step t, the first stack contains the sequence of video clips yet to be processed $V_t, V_{t+1}, \ldots, V_N$. The direction of the LSTM goes from V_N to V_t, which allows the information to flow from the future clips to the current clip. We refer to this LSTM network as the video stack and denote its hidden state as h_t^V. Similarly, the text stack contains the sentence sequence yet to be processed: $S_t, S_{t+1}, \ldots, S_M$. Its hidden state is h_t^S.

The third stack is the action stack, which stores all alignment actions performed in the past. The actions are denoted as $A_{t-1}, \ldots, A_1$ and are encoded as one-hot vectors $a_{t-1}, \ldots, a_1$. The reason for including this stack is to capture patterns in the historic actions. Different from the first two stacks, the information flows from the first action to the immediate past with the last hidden state being h_{t-1}^A.

The fourth stack is the matched stack, which contains only the texts and clips that are matched previously and places the last matched content at the top of the stack. We denote this sequence as $R_1, \ldots, R_L$. Similar to the action stack, the information flows from the past to the present. In this paper, we consider the case where one sentence s_i can match with multiple video clips $v_1, \ldots, v_K$. Since the matched video clips are probably similar in content, we perform mean pooling over the video features $v_i = \sum_j^K v_j / K$. The input to the LSTM unit is hence the concatenation of the two modalities $r_i = [s_i, v_i]$. The last hidden state of the matched stack is h_{t-1}^M.

Alignment Action Prediction. At every time step, the state of the four stacks is $\Psi_t = (V_{t^+}, S_{t^+}, A_{(t-1)^-}, R_{1^+})$, where we use the shorthand X_{t^+} for the sequence $X_t, X_{t+1}, \ldots$ and similarly for X_{t^-}. Ψ_t can be approximately represented by the LSTM hidden states. Thus, the conditional probability of alignment action A_t at time t is

$$P(A_t|\Psi_t) = P(A_t|h_t^V, h_t^S, h_{t-1}^A, h_{t-1}^M) \qquad (4)$$

The above computation is implemented as a softmax operation after two fully connected layers with ReLU activation on top of the concatenated state $\psi_t = [h_t^V, h_t^S, h_{t-1}^A, h_{t-1}^M]$. In order to compute the alignment of entire sequences, we apply the chain rule.

$$P(A_1, \ldots, A_N|\mathcal{V}, \mathcal{S}) = \prod_{t=1}^{N} P(A_t|A_{(t-1)^-}, \Psi_t) \qquad (5)$$

The probability can be optimized greedily by always choosing the most probable action or using beam search. The classification is trained in a supervised manner. From a ground truth alignment of two sequences, we can easily derive a correct sequence of actions, which are used in training. In the infrequent case when more than one correct action sequence exist, one is randomly picked. The training objective is to minimize the cross-entropy loss at every time step.

	Video Stack	**Text Stack**	**Matched Stack**	**Action Stack**
Initial	ⓐⓑⓒ	①②③		
Pop Clip	ⓑⓒ	①②③		PC
Pop Sent	ⓐⓑⓒ	②③		PS
Match	ⓑⓒ	②③	[ⓐ①]	M
Match-Retain-C	ⓐⓑⓒ	②③	[ⓐ①]	MRC
Match-Retain-S	ⓑⓒ	①②③	[ⓐ①]	MRS

Table 2: The basic action inventory and their effects on the stacks. Square brackets indicate matched elements.

Alignment Actions. We propose five basic alignment actions that together handle the alignment of two sequences with unmatched elements and one-to-many matching. The actions include *Pop Clip* (PC), *Pop Sentence* (PS), *Match* (M), *Match-Retain Clip* (MRC), and *Match-Retain Sentence* (MRS). Table 2 provides a summary of their effects.

The Pop Clip action removes the top element, V_t, from the video stack. This is desirable when V_t does not match any element in the text stack. Analogously, the *Pop Sentence* action removes the top element in the text stack, S_t. The Match action removes both V_t and S_t, matches them, and pushes them to the matched stack. The actions Match-Retain Clip and Match-Retain Sentence are only used for one-to-many correspondence. When many sentences can be matched with one video clip, the Match-Retain Clip action pops S_t, matches it with V_t and pushes the pair to the matched stack, but V_t stays on the video stack for the next possible sentence. To pop V_t, the Pop Clip action must be used. The Match-Retain Sentence action is similarly defined. In this formulation, matching is always between elements at the top of the stacks.

It is worth noting that the five actions do not have to be used together. A subset can be picked based on knowledge about the sequences being matched. For example, for one-to-one matching, if we know some clips may not match any sentences, but every sentence have at least one matching clip, we only need Pop Clip and Match. Alternatively, consider a one-to-many scenario where (1) one sentence can match multiple video clips, (2) some clips are unmatched, and (3) every sentence has at least one matching clip. We need only the subset Pop Clip, Pop Sentence, and Match-

Retain Sentence. It is desirable to choose as few actions as possible, because it simplifies training and reduces the branching factor during inference.

Discussion. The utility of the action stack becomes apparent in the one-to-many setting. As discussed earlier, to encode an element R_i in the matched stack, features from different video clips are mean-pooled. As a result, if the algorithm needs to learn a constraint on how many clips can be merged together, features from the matched stack may not be effective, but features from action stack would carry the necessary information. The alignment actions discussed in the above section allow monotonic matching for two sequences, which is the focus of this paper and experiments. We discuss extensions that allow multi-sequence matching as well as non-monotonic matching in Appendix A.

4. Experimental Evaluation

We evaluate NeuMATCH on semi-synthetic and real datasets, including a newly annotated, real-world YouTube Movie Summaries (YMS) dataset. Table 3 shows the statistics of the datasets used.

4.1. Datasets

We create the datasets HM-1 and HM-2 based on the LSMDC data [35], which contain matched clip-sentence pairs. The LSMDC data contain movie clips and very accurate textual descriptions, which are originally intended for the visually impaired. We generate video and textual sequences in the following way: First, video clips and their descriptions in the same movie are collected sequentially, creating the initial video and text sequences. For HM-1, we randomly insert video clips from other movies into each video sequence. In order to increase the difficulty of alignment and to make the dataset more realistic, we select confounding clips that are similar to the neighboring clips. After randomly choosing an insertion position, we sample 10 video clips and select the most similar to its neighboring clips, using the pre-trained similarity metric (Section 3.1). An insertion position can be 0-3 clips away from the last insertion. For HM-2, we randomly delete sentences from the collected text sequences. A deletion position is 0-3 sentences from the last deletion. At this point, HM-1 and HM-2 does not require one-to-many matching, which is used to test the 2-action NeuMATCH model. To allow one-to-many matching, we further randomly split every video clip into 1-5 smaller clips.

YMS dataset. We create the YMS dataset from the YouTube channels *Movie Spoiler Alert* and *Movies in Minutes*, where a narrator orally summarizes movies alongside clips from the actual movie. Two annotators transcribed the audio and aligned the narration text with video clips. The YMS dataset is the most challenging for several reasons: (1) The sequences are long. On average, a video sequence contains 161.5 clips and a textual sequence contains 58.2 sentences. (2) A sentence can match a long sequence of (up to 45) video clips. (3) Unlike LSMDC, YMS contains rich textual descriptions that are intended for storytelling; they are not always faithful descriptions of the video, which makes YMS a challenging benchmark.

	HM-1	**HM-2**	**YMS**
# words	4,196,633	4,198,021	54,326
# sent.	458,557	458,830	5,470
# avg. words/sent.	9.2	9.1	9.5
# clips	1,788,056	1,788,056	15,183
# video	22,945	22,931	94
# avg clips/video	77.9	77.9	161.5
# avg sent./video	20.0	20.0	58.2
# clip/sent. (mean(var))	2.0(0.33)	2.0(0.33)	2.6(8.8)

Table 3: Summary statistics of the datasets.

	HM-1				**HM-2**			
	MD	*CTW*	*DTW*	***Ours***	*MD*	*CTW*	*DTW*	***Ours***
clips	6.4	13.4	13.3	**69.7**	2.5	12.9	13.0	**40.6**
sents.	15.8	21.3	41.7	**58.6**	15.6	25.1	34.2	**43.7**

Table 4: Accuracy of clips and sentences for the 2-action model. Datasets require the detection of *null* clips.

4.2. Performance Metrics

For one-to-one matching, we measure the matching accuracy, or the percentage of sentences and video clips that are correctly matched or correctly assigned to *null*. For one-to-many matching, where one sentence can match multiple clips, we cannot use the same accuracy for sentences. Instead, we turn to the Jaccard Index, which measures the overlap between the predicted range and the ground truth of video clips using the intersection over union (IoU).

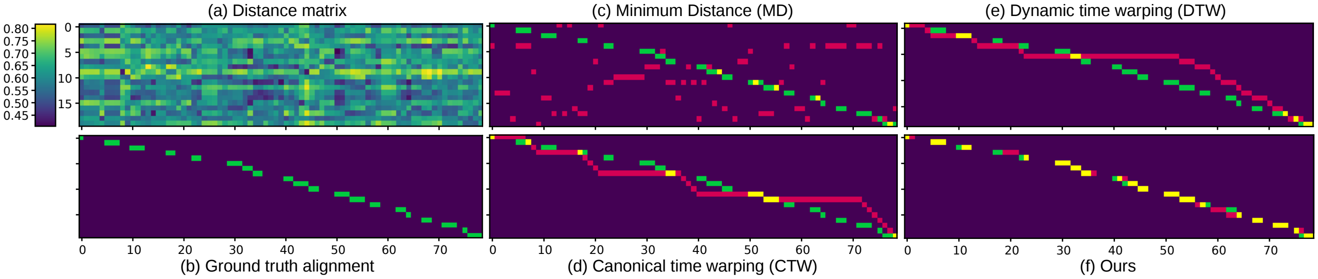
4.3. Baselines

We create three baselines, Minimum Distance (MD), Dynamic Time Warping (DTW), and Canonical Time Warping (CTW). All baselines use the same jointly trained language-visual neural network encoders (Section 3.1), which are carefully trained and exhibit strong performance. Due to space constraints, we discuss implementation details in the supplementary material.

The MD method matches the most similar clip-sentence pairs which have the smallest distance compared to the others. We artificially boost this baseline using specific optimization for the two accuracy measures. For evaluation on video clips, we match every clip with the most similar sentence, but if the distance is greater than the threshold 0.7, we consider the clip to be unmatched (i.e., a *null clip*). For

	HM-0				HM-1				HM-2				YMS			
	MD	*CTW*	*DTW*	***Ours***	*MD*	*CTW*	*DTW*	***Ours***	*MD*	*CTW*	*DTW*	***Ours***	*MD*	*CTW*	*DTW*	***Ours***
clips	20.7	26.3	50.6	**63.1**	10.5	6.8	17.6	**65.0**	10.6	6.9	18.0	**37.7**	4.0	5.0	10.3	**12.0**
sents IoU	23.0	25.4	42.8	**55.3**	5.7	7.3	18.4	**44.1**	9.0	7.6	18.9	**20.0**	2.4	3.6	7.5	**10.4**

Table 5: Alignment performance for 3-action model given in percentage (%) over all data. Datasets HM-1, HM-2, and YMS require the detection of null clips and one-to-many matchings of the sentences. HM-0 only requires one-to-many matching of sentences.



(a) Distance matrix

(c) Minimum Distance (MD)

(e) Dynamic time warping (DTW)

(b) Ground truth alignment

(d) Canonical time warping (CTW)

(f) Ours

Figure 4: An alignment problem from HM-2 and the results. The vertical and horizontal axes represent the text sequence (sentences) and video sequence (clips) respectively. Green, red and yellow respectively represent the ground-truth alignment, the predicted alignment, and the intersection of two.

sentence accuracy, we match every sentence with the most similar clip and do not assign *null* sentences.

DTW computes the optimal path on the distance matrix. It uses the fact that the first sentence is always matched with the first clip, and the last sentence is always matched to the last clip, so the shortest path is between the upper left corner and lower right corner of the distance matrix. Note this is a constraint that NeuMATCH is not aware of. In order to handle null clips, we make use of the threshold again. In the case that one sentence is matched with several clips, the clips whose distances with the sentence are above the threshold will be assigned to null. We manually tuned the threshold to maximize the performance of all baselines. For CTW, we adopt source code provided in [59] with the same assignment method as DTW.

4.4. Results and Discussion

Tables 4 and 5 show the performance under one-to-one and one-to-many scenarios, respectively. On the one-to-one versions of the datasets HM-1 and HM-2, NeuMATCH demonstrates considerable improvements over the best baselines. It improves clip accuracy by 56.3 and 27.6 percentage points and improves sentence accuracy by 16.9 and 9.5 points. Unlike CTW and DTW, NeuMATCH does not have a major gap between clip and sentence performance.

On the one-to-many versions of HM-1 and HM-2, as well as the YMS dataset, NeuMATCH again shows superior performance over the baselines. The advantage over the best baselines is 47.4, 19.7, and 1.7 points for clip accuracy, and 25.7, 1.1, and 2.9 for sentence IoU. Interestingly, NeuMATCH performs better on HM-1 than HM-2, but the other baselines are largely indifferent between the two datasets. This is likely due to NeuMATCH's ability to extract information from the matched stack. Since HM-1 is created by inserting random clips into the video sequence, the features of the inserted video clip match surrounding clips, but other aspects such as cinematography style may not match. This makes HM-1 easier for NeuMATCH because it can compare the inserted clip with those in the matched stack and detect style differences. It is worth noting that different cinematographic styles are commonly used to indicate memories, illusions, or imaginations. Being able to recognize such styles can be advantageous for understanding complex narrative content.

To further investigate NeuMATCH's performance without null clips, we additionally create a one-to-many dataset, HM-0, by randomly dividing every video clip into 1-to-5 smaller clips. Although NeuMATCH's advantage is reduced on HM-0, it's still substantial (12.5 points on both measures), showing that the performance gains are not solely due to the presence of null clips.

As we expect, the real-world YMS dataset is more difficult than HM-1 and HM-2. Still, we have a relative improvement of 17% on clip accuracy and 39% on sentence IoU over the closest DTW baseline. We find that NeuMATCH consistently surpasses conventional baselines across all experimental conditions. This clearly demonstrates NeuMATCH's ability to identify alignment from heterogenous video-text inputs that are challenging to understand computationally.

As a qualitative evaluation, Figure 4 shows an alignment example. The ground alignment goes from the top left (the first sentence and the first clip) to the bottom right (the last

	HM-1		HM-2	
	clips	sent. IoU	clips	sent. IoU
No Act&Hist	47.3	21.8	11.8	1.6
No Action	49.9	23.0	29.6	16.1
No History	57.6	33.4	28.3	17.0
No Input LSTMs	54.8	24.6	27.9	8.3
NeuMATCH	**65.0**	**44.1**	**37.7**	**20.0**

Table 6: Performance of ablated models in the one-to-many setting (3-action model).

sentence and the last clip). Dots in green, red, and yellow represent the ground truth alignment, the predicted alignment, and the intersection of the two, respectively. In the ground truth path (e), some columns does not have any dots because those clips are not matched to anything. As shown in (a), the distance matrix does not exhibit any clear alignment path. Therefore, MD, which uses only the distance matrix, performs poorly. The time warping baselines in (c) and (d) also notably deviate from the correct path, whereas NeuMATCH is able to recover most of the ground-truth alignment. For more alignment examples, we refer interested readers to the supplementary material.

4.5. Ablation Study

In order to understand the benefits of the individual components of NeuMATCH, we perform an ablated study where we remove one or two LSTM stacks from the architecture. The model *No Act&Hist* lacks both the action stack and the matched stack in the alignment network. That is, it only has the text and the video stacks. The second model *No Action* and the third model *No History* removes the action stack and the matched stack, respectively. In the last model *No Input LSTM*, we directly feed features of the video clip and the sentence at the tops of the respective stacks into the alignment network. That is, we do not consider the influence of future input elements.

Table 6 shows the performance of four ablated models in the one-to-many setting. The four ablated models perform substantially worse than the complete model. This confirms our intuition that both the history and the future play important roles in sequence alignment. We conclude that all four LSTM stacks contribute to NeuMATCH's superior performance.

5. Conclusions

In this paper, we propose NeuMATCH, an end-to-end neural architecture aimed at heterogeneous multi-sequence alignment, focusing on alignment of video and textural data. Alignment actions are implemented in our network as data moving operations between LSTM stacks. We show that this flexible architecture supports a variety of alignment tasks. Results on semi-synthetic and real-world datasets and multiple different settings illustrate superiority of this model over popular traditional approaches based on time warping. An ablation study demonstrates the benefits of using rich context when making alignment decisions.

Acknowledgement. Pelin Dogan was partially funded by the SNF grant 200021_153307/1.

A. Extensions to Multiple Sequences and Non-monotonicity

The basic action inventory tackles the alignment of two sequences. The alignment of more than two sequences simultaneously, like video, audio, and textual sequences, requires an extension of the action inventory. To this end, we introduce a parameterized *Match-Retain* action. For three sequences, the parameters are a 3-bit binary vector where 1 indicate the top element from this sequence is being matched and 0 otherwise. Table 7 shows one example using the parameterized Match-Retain. For instance, to match the top elements from Sequence A and B, the action is Match-Retain (110). The parameters are implemented as three separate binary predictions.

The use of parameterized actions further enables non-monotonic matching between sequences. In all previous examples, matching only happens between the stack tops. Non-monotonic matching is equivalent to allowing stack top elements to match with any element on the matched stack. We propose a new parameterized action *Match-With-History*, which has a single parameter q that indicates position on the matched stack. To deal with the fact that the matched stack has a variable length, we adopt the indexing method from Pointer Networks [49]. The probability of choosing the i^{th} matched element r_i is

$$P(q = i|\Psi_t) = \frac{\exp(f(\psi_t, r_i))}{\sum_{j=0}^{L} \exp(f(\psi_t, r_j))} \tag{6a}$$

$$f(\psi_t, r_i) = v^\top \tanh\left(W_q \begin{bmatrix}\psi_t \\ r_i\end{bmatrix}\right) \tag{6b}$$

where the matrix W_q and vector v are trainable parameters and L is the length of the matched stack.

	Seq A	Seq B	Seq C	Matched Stack
Initial	ⓐⓑⓒ	①②③	ⓧⓨⓩ	
1. M-R(110)	ⓐⓑⓒ	①②③	ⓧⓨⓩ	[ⓐ①]
2. Pop A	ⓑⓒ	①②③	ⓧⓨⓩ	[ⓐ①]
3. Pop B	ⓑⓒ	②③	ⓧⓨⓩ	[ⓐ①]
4. M-R(011)	ⓑⓒ	②③	ⓧⓨⓩ	[②ⓧ][ⓐ①]

Table 7: An example action sequence for aligning three sequences.

References

[1] P. Anderson, X. He, C. Buehler, D. Teney, M. Johnson, S. Gould, and L. Zhang. Bottom-up and top-down attention for image captioning and VQA. *arXiv preprint arXiv:1707.07998*, 2017.

[2] S. Antol, A. Agrawal, J. Lu, M. Mitchell, D. Batra, C. L. Zitnick, and D. Parikh. VQA: Visual question answering. In *ICCV*, pages 2425–2433, 2015.

[3] R. Barzilay and L. Lee. Learning to paraphrase: an unsupervised approach using multiple-sequence alignment. In *NAACL*, 2003.

[4] D. J. Berndt and J. Clifford. Using dynamic time warping to find patterns in time series. In *KDD workshop*, 1994.

[5] P. Bojanowski, R. Lajugie, E. Grave, F. Bach, I. Laptev, J. Ponce, and C. Schmid. Weakly-supervised alignment of video with text. In *ICCV*, pages 4462–4470, 2015.

[6] S. R. Bowman, G. Angeli, C. Potts, , and C. D. Manning. A large annotated corpus for learning natural language inference. *EMNLP*, 2015.

[7] Y. Caspi and M. Irani. A step towards sequence-to-sequence alignment. In *CVPR*, 2000.

[8] A. Conneau, D. Kiela, H. Schwenk, L. Barrault, and A. Bordes. Supervised learning of universal sentence representations from natural language inference data. *arXiv preprint arXiv:1705.02364*, 2017.

[9] T. Cour, C. Jordan, E. Miltsakaki, and B. Taskar. Movie/script: Alignment and parsing of video and text transcription. *ECCV*, pages 158–171, 2008.

[10] Z. Deng, A. Vahdat, H. Hu, and G. Mori. Structure inference machines: Recurrent neural networks for analyzing relations in group activity recognition. In *CVPR*, pages 4772–4781, 2016.

[11] P. Dogan, M. Gross, and J.-C. Bazin. Label-based automatic alignment of video with narrative sentences. In *ECCV Workshops*, pages 605–620, 2016.

[12] J. Donahue, L. Anne Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. Darrell. Long-term recurrent convolutional networks for visual recognition and description. In *CVPR*, 2015.

[13] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. In *ICML*, pages 647–655, 2014.

[14] C. Dyer, M. Ballesteros, W. Ling, A. Matthews, and N. A. Smith. Transition-based dependency parsing with stack long short-term memory. 2015.

[15] M. Everingham, J. Sivic, and A. Zisserman. Hello! My name is... Buffy—automatic naming of characters in TV video. In *BMVC*, 2006.

[16] H. Fang, S. Gupta, F. Iandola, R. K. Srivastava, L. Deng, P. Dollár, J. Gao, X. He, M. Mitchell, J. C. Platt, et al. From captions to visual concepts and back. In *CVPR*, pages 1473–1482, 2015.

[17] A. Farhadi, M. Hejrati, M. A. Sadeghi, P. Young, C. Rashtchian, J. Hockenmaier, and D. Forsyth. Every picture tells a story: Generating sentences from images. In *ECCV*, pages 15–29, 2010.

[18] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016.

[19] M. Hodosh, P. Young, and J. Hockenmaier. Framing image description as a ranking task: Data, models and evaluation metrics. *Journal of Artificial Intelligence Research*, 47:853–899, 2013.

[20] M. Honnibal and M. Johnson. An improved non-monotonic transition system for dependency parsing. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1373–1378, 2015.

[21] A. Karpathy and L. Fei-Fei. Deep visual-semantic alignments for generating image descriptions. In *CVPR*, pages 3128–3137, 2015.

[22] R. Kiros, R. Salakhutdinov, and R. S. Zemel. Unifying visual-semantic embeddings with multimodal neural language models. *Transactions of the Association for Computational Linguistics*, 2014.

[23] R. Kiros, Y. Zhu, R. R. Salakhutdinov, R. Zemel, R. Urtasun, A. Torralba, and S. Fidler. Skip-thought vectors. In *NIPS*, pages 3294–3302, 2015.

[24] C. Kong, D. Lin, M. Bansal, R. Urtasun, and S. Fidler. What are you talking about? text-to-image coreference. In *CVPR*, pages 3558–3565, 2014.

[25] R. Krishna, K. Hata, F. Ren, L. Fei-Fei, and J. C. Niebles. Dense-captioning events in videos. In *ICCV*, 2017.

[26] D. Lin, S. Fidler, C. Kong, and R. Urtasun. Visual semantic search: Retrieving videos via complex textual queries. In *CVPR*, pages 2657–2664, 2014.

[27] A. Löytynoja and N. Goldman. An algorithm for progressive multiple alignment of sequences with insertions. *Proceedings of the National academy of sciences of the United States of America*, 102(30):10557–10562, 2005.

[28] J. Malmaud, J. Huang, V. Rathod, N. Johnston, A. Rabinovich, and K. Murphy. What's cookin'? interpreting cooking videos using text, speech and vision. *arXiv preprint*, 2015.

[29] J. Mao, W. Xu, Y. Yang, J. Wang, Z. Huang, and A. Yuille. Deep captioning with multimodal recurrent neural networks (m-rnn). *arXiv preprint*, 2014.

[30] T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[31] Y. Pan, T. Mei, T. Yao, H. Li, and Y. Rui. Jointly modeling embedding and translation to bridge video and language. In *CVPR*, pages 4594–4602, 2016.

[32] J. Pennington, R. Socher, and C. D. Manning. GloVe: Global vectors for word representation. In *EMNLP*, pages 1532–1543, 2014.

[33] J. Prokić, M. Wieling, and J. Nerbonne. Multiple sequence alignments in linguistics. In *EACL Workshop on Language Technology and Resources for Cultural Heritage, Social Sciences, Humanities, and Education*, pages 18–25, 2009.

[34] M. Ranzato, A. Szlam, J. Bruna, M. Mathieu, R. Collobert, and S. Chopra. Video (language) modeling: a baseline for generative models of natural videos. *arXiv preprint arXiv:1412.6604*, 2014.

[35] A. Rohrbach, M. Rohrbach, N. Tandon, and B. Schiele. A dataset for movie description. In *CVPR*, pages 3202–3212, 2015.

[36] F. Sadeghi, S. K. Kumar Divvala, and A. Farhadi. Viske: Visual knowledge extraction and question answering by visual verification of relation phrases. In *CVPR*, pages 1456–1464, 2015.

[37] P. Sankar, C. Jawahar, and A. Zisserman. Subtitle-free movie to script alignment. In *BMVC*, 2009.

[38] S. Sheinfeld, Y. Gingold, and A. Shamir. Video summarization using causality graphs. In *HCOMP Workshop on Human Computation for Image and Video Analysis*, 2016.

[39] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[40] R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. The MIT Press, second (complete draft) edition, 2017.

[41] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *CVPR*, pages 1–9, 2015.

[42] M. Tapaswi, M. Bäuml, and R. Stiefelhagen. Story-based video retrieval in tv series using plot synopses. In *International Conference on Multimedia Retrieval*, page 137. ACM, 2014.

[43] M. Tapaswi, M. Bauml, and R. Stiefelhagen. Book2movie: Aligning video scenes with book chapters. In *CVPR*, pages 1827–1835, 2015.

[44] A. Torabi, N. Tandon, and L. Sigal. Learning language-visual embedding for movie understanding with natural-language. *arXiv preprint arXiv:1609.08124*, 2016.

[45] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri. Learning spatiotemporal features with 3d convolutional networks. In *ICCV*, pages 4489–4497, 2015.

[46] I. Vendrov, R. Kiros, S. Fidler, and R. Urtasun. Order-embeddings of images and language. *arXiv preprint arXiv:1511.06361*, 2015.

[47] S. Venugopalan, M. Rohrbach, J. Donahue, R. Mooney, T. Darrell, and K. Saenko. Sequence to sequence-video to text. In *ICCV*, pages 4534–4542, 2015.

[48] S. Venugopalan, H. Xu, J. Donahue, M. Rohrbach, R. Mooney, and K. Saenko. Translating videos to natural language using deep recurrent neural networks. *arXiv preprint arXiv:1412.4729*, 2014.

[49] O. Vinyals, M. Fortunato, and N. Jaitly. Pointer networks. In *NIPS*, pages 2692–2700, 2015.

[50] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan. Show and tell: A neural image caption generator. In *CVPR*, pages 3156–3164, 2015.

[51] H. Wang, A. Kläser, C. Schmid, and C.-L. Liu. Action recognition by dense trajectories. In *CVPR*, pages 3169–3176, 2011.

[52] B. Xu, Y. Fu, Y. G. Jiang, B. Li, and L. Sigal. Heterogeneous knowledge transfer in video emotion recognition, attribution and summarization. *IEEE Transactions on Affective Computing*, 2017.

[53] H. Xu and K. Saenko. Ask, attend and answer: Exploring question-guided spatial attention for visual question answering. In *ECCV*, pages 451–466, 2016.

[54] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhudinov, R. Zemel, and Y. Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *ICML*, pages 2048–2057, 2015.

[55] R. Xu, C. Xiong, W. Chen, and J. J. Corso. Jointly modeling deep video and compositional text to bridge vision and language in a unified framework. In *AAAI*, volume 5, page 6, 2015.

[56] L. Yao, A. Torabi, K. Cho, N. Ballas, C. Pal, H. Larochelle, and A. Courville. Describing videos by exploiting temporal structure. In *ICCV*, pages 4507–4515, 2015.

[57] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson. How transferable are features in deep neural networks? In *NIPS*, pages 3320–3328, Cambridge, MA, USA, 2014. MIT Press.

[58] H. Yu, J. Wang, Z. Huang, Y. Yang, and W. Xu. Video paragraph captioning using hierarchical recurrent neural networks. In *CVPR*, pages 4584–4593, 2016.

[59] F. Zhou and F. De la Torre. Generalized canonical time warping. *PAMI*, 38(2):279–294, 2016.

[60] Y. Zhu, R. Kiros, R. Zemel, R. Salakhutdinov, R. Urtasun, A. Torralba, and S. Fidler. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *ICCV*, pages 19–27, 2015.

[61] B. Zoph, V. Vasudevan, J. Shlens, and Q. V. Le. Learning transferable architectures for scalable image recognition. *arXiv preprint arXiv:1707.07012*, 2017.