Discriminative Unsupervised Alignment of Natural Language Instructions with Corresponding Video Segments

Iftekhar Naim[1], **Young Chol Song**[1], **Qiguang Liu**[1], **Liang Huang**[2], **Henry Kautz**[1], **Jiebo Luo**[1] and **Daniel Gildea**[1]
[1]Department of Computer Science, University of Rochester, Rochester, NY 14627
[2]Queens College & Graduate Center, City University of New York, Flushing, NY 11367

Abstract

We address the problem of automatically aligning natural language sentences with corresponding video segments without any direct supervision. Most existing algorithms for integrating language with videos rely on hand-aligned parallel data, where each natural language sentence is manually aligned with its corresponding image or video segment. Recently, fully unsupervised alignment of text with video has been shown to be feasible using hierarchical generative models. In contrast to the previous generative models, we propose three latent-variable discriminative models for the unsupervised alignment task. The proposed discriminative models are capable of incorporating domain knowledge, by adding diverse and overlapping features. The results show that discriminative models outperform the generative models in terms of alignment accuracy.

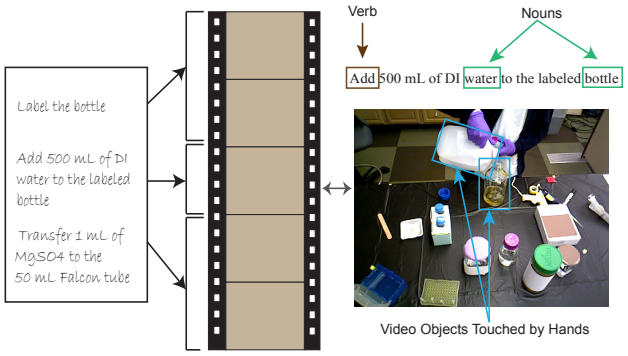

Figure 1: The proposed discriminative learning algorithm aligns protocol sentences to corresponding video frames. We incorporate features that can learn the co-occurrences of nouns and verbs in the sentences with the objects in the video.

1 Introduction

Learning to integrate natural language descriptions with video events is attracting increasing attention in the natural language processing and computer vision communities. The Grounded Language Learning task aims to map the meaning of natural language expressions to their corresponding referents in videos (e.g., objects, actions, and events) without any dictionary. Most existing grounded language learning algorithms are either supervised or weakly-supervised. During the training stage, they assume each video is pre-segmented to chunks of short duration, and each video segment is manually aligned with a natural language sentence that describes that segment. Manually aligning each video segment with a sentence is tedious, especially for long videos. Therefore, it is desirable to automatically align video segments with their corresponding natural language sentences without direct supervision.

Recently, Naim et al. (2014) proposed an unsupervised learning algorithm for automatically aligning sentences in a document with corresponding video segments. Given a sequence of natural language instructions and an unaligned video recording of a person following these instructions, a hierarchical generative model was applied to align each instruc-

tion to its corresponding video segment, and to align nouns in each instruction to their corresponding objects in the video. We extend this generative alignment framework by applying several discriminative models with latent variables. Discriminative models are attractive as they can easily incorporate domain knowledge by adding many diverse, overlapping, and complex features. By incorporating a large number of features and regularizing their weights properly, discriminative models have been shown to outperform generative models in many natural language processing tasks (Collins, 2002; Dyer et al., 2011; Yu et al., 2013).

Similar to Naim et al. (2014), we applied our algorithm to align the natural language instructions for biological experiments in "wet laboratories" with recorded videos of people performing these experiments. Typically, each wetlab experiment has a protocol written in natural language, describing the sequence of steps necessary for that experiment. However, these instructions are often incomplete, and do not spell out implicit assumptions and knowledge, causing the results to be difficult to reproduce (Begley and Ellis, 2012). Given a set of such wetlab experiment protocols and associated videos, our initial goal is to infer the correct alignment between the steps mentioned in the protocol and corresponding video segments in which a person performs these steps (Figure 1). The aligned and segmented output of the system described in this paper can eventually be used to learn detailed visual models of correctly performed activities and to identify experimental anomalies.

In this paper, we apply three latent discriminative learning algorithms: latent conditional random field (LCRF), latent structured perceptron (LSP), and latent structured support vector machine (LSSVM) for unsupervised alignment of video with text. We show that discriminative models outperform the existing generative models by incorporating diverse features. While the previous models only considered the mappings of nouns to blobs, and ignored verbs, we incorporated the co-occurrences of verbs with blobs as features in our model. Finally, we propose a constrained variant of the standard LSP and LSSVM update rule, which provided better alignment accuracy and more stable convergence on our datasets.

2 Background Research

2.1 Unsupervised Grounded Language Learning

Most existing grounded language learning algorithms for integrating language with vision rely on either a fully supervised (Kollar et al., 2010; Matuszek et al., 2012) or a weakly supervised training stage (Yu and Ballard, 2004; Kate and Mooney, 2007; Krishnamurthy and Kollar, 2013; Yu and Siskind, 2013; Krishnamoorthy et al., 2013; Rohrbach et al., 2013; Tellex et al., 2013). The fully supervised methods assume that each sentence in the training data is manually paired with the corresponding image or video segment, and furthermore, each word or phrase in a sentence is already mapped to its corresponding blob or action in the image or video segment. Given the detailed annotations, these methods train a set of classifiers to recognize perceptual representations for commonly used words or phrases. After the initial fully supervised training stage, these methods can learn the meaning of new words as they are encountered. Such detailed supervision is difficult to obtain, and as a result most of the recent grounded language learning algorithms rely on weaker supervision (Krishnamurthy and Kollar, 2013; Yu and Siskind, 2013; Krishnamoorthy et al., 2013; Rohrbach et al., 2013; Tellex et al., 2013), where each image or video frame is manually paired with corresponding sentence, but the mapping between objects and words is not provided, and instead learned and inferred automatically as latent variables. Manually pairing each video segment or image frame with the corresponding sentence can be tedious, especially for long videos. Furthermore, these methods can be relatively difficult to extend to new domains, as this may require collecting new annotated data.

Recently, Naim et al. (2014) proposed a fully unsupervised approach for aligning wetlab experiment videos with associated text protocols, without any direct supervision. They proposed a hierarchical generative model to infer the alignment between each video segment with corresponding protocol sentence, and also the mapping of each blob with corresponding noun in that sentence. First, it models the generation of each video segment from one of the sentences in the protocol using a Hidden

Markov Model (HMM) (Rabiner, 1989; Vogel et al., 1996). Next, each tracked object or blob in a video segment is generated from one of the nouns in the corresponding sentence using IBM Model 1 (Brown et al., 1993), a generative model frequently used in machine translation. The IBM Model 1 probabilities are incorporated as emission probabilities in HMM. The transition probabilities are parameterized using the jump size, i.e., the difference between the alignments of two consecutive video segments. They also extended IBM Model 1 by introducing latent variables for each noun, allowing some of the non-object nouns to be unobserved in the video. While the alignment results are encouraging, and show that unsupervised alignment is feasible, they considered the mappings between nouns and blobs only, and ignored the verbs and other relations in the sentences. Moreover, incorporating domain knowledge is not straightforward in these generative models.

2.2 Discriminative Word Alignment

In machine translation, alignment of the words in source language with the words in target language has traditionally been done using the IBM word alignment models (Brown et al., 1993), which are generative models, and typically trained using Expectation Maximization (Dempster et al., 1977). Early attempts (Blunsom and Cohn, 2006; Taskar et al., 2005) towards discriminative word alignment relied on supervised hand-aligned parallel corpora. Dyer et al. (2011) first applied a latent variable conditional random field (LCRF) to perform unsupervised discriminative word alignment. They treated the words' alignments as latent variables, and formulated the task as predicting the target sentence, given the source sentence. We apply similar latent variable discriminative models for unsupervised alignment of sentences with video segments.

3 Problem Formulation and Notations

The input to our system is a dataset containing N pairs of observations $\{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^{N}$, where $\mathbf{x}_i$ represents the i^{th} experiment protocol, and $\mathbf{y}_i$ represents a video of a person carrying out the instructions in that protocol. The protocols are not necessarily unique, as we have multiple videos of different people carrying out the same protocol.

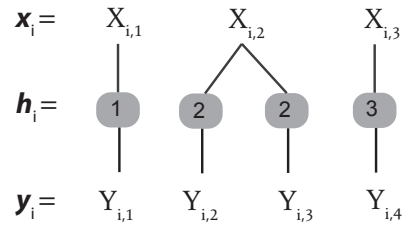

Figure 2: Each $X_{i,m}$ is a sentence in the protocol, consisting of the nouns and verbs in the sentence, and each $Y_{i,n}$ is a video chunk represented by the set of blobs touched by hands in that chunk. The alignment $\mathbf{h}_i = [1, 2, 2, 3]$ maps each video chunk to the corresponding sentence.

We apply similar data preprocessing as Naim et al. (2014). First, we parse each protocol sentence using the two-stage Charniak-Johnson parser (Charniak and Johnson, 2005), and extract the head nouns and verbs from each sentence. Let m_i be the number of sentences in the protocol $\mathbf{x}_i$. We represent $\mathbf{x}_i$ as a sequence of sets $\mathbf{x}_i = [X_{i,1}, \ldots, X_{i,m_i}]$, where $X_{i,m}$ is the set of nouns and verbs in the m^{th} sentence of $\mathbf{x}_i$. Each video $\mathbf{y}_i$ is segmented into a sequence of chunks, each one second long. For each video chunk, we determine the set of objects touched by the participant's hands using automated image segmentation and tracking. We ignore the chunks over which no object is touched by a hand. Let n_i be the number of chunks in $\mathbf{y}_i$. We represent the video $\mathbf{y}_i$ as a sequence of sets: $\mathbf{y}_i = [Y_{i,1}, \ldots, Y_{i,n_i}]$, one for each video chunk, where $Y_{i,n}$ is the set of objects or blobs touched by hands in the n^{th} chunk of $\mathbf{y}_i$. If V_Y is the set of all blobs in the videos, then $Y_{i,n} \subseteq V_Y$.

Our goal is to learn the alignment $\mathbf{h}_i$ between the sentences in $\mathbf{x}_i$ with their corresponding video chunks in $\mathbf{y}_i$ (Figure 2). Formally, $\mathbf{h}_i[n] \in \{1, \ldots, m_i\}$, for $1 \leq n \leq n_i$, where $\mathbf{h}_i[n] = m$ indicates that the video segment $Y_{i,n}$ is aligned to the protocol sentence $X_{i,m}$.

4 Discriminative Alignment

To formulate the alignment problem as a discriminative learning task, we assume the text sequence $\mathbf{x}_i$ as the observed input, and the video sequence $\mathbf{y}_i$ as the output sequence that we aim to predict. Since the alignments are unknown, we treat them

as latent variables. Let $\mathbf{h}_i$ be the hidden alignment vector for an observation pair $(\mathbf{x}_i, \mathbf{y}_i)$. The feature function $\mathbf{\Phi}(\mathbf{x}_i, \mathbf{y}_i, \mathbf{h}_i)$ maps the input observation $(\mathbf{x}_i, \mathbf{y}_i)$, and their latent alignment vector $\mathbf{h}_i$ to a d-dimensional feature vector. Our goal is to learn the weights $\mathbf{w} \in \mathbb{R}^d$ for these features.

4.1 Latent Variable Conditional Random Field

Given a text sequence $\mathbf{x}_i$ and a video sequence $\mathbf{y}_i$ with lengths $|\mathbf{x}_i| = m_i$ and $|\mathbf{y}_i| = n_i$, the conditional probability of the video sequence is:

$$\begin{aligned} p(\mathbf{y}_i|\mathbf{x}_i) &= p(\mathbf{y}_i, n_i|\mathbf{x}_i) \\ &= p(\mathbf{y}_i|\mathbf{x}_i, n_i)\, p(n_i|\mathbf{x}_i) \end{aligned} \quad (1)$$

Since we only aim to learn the alignments given $(\mathbf{x}_i, \mathbf{y}_i)$, we ignore the length probability $p(n_i|\mathbf{x}_i)$, and consider only the first term:

$$p(\mathbf{y}_i|\mathbf{x}_i, n_i) = \sum_{\mathbf{h}_i} p(\mathbf{y}_i, \mathbf{h}_i|\mathbf{x}_i, n_i) \quad (2)$$

We model the conditional probability $p(\mathbf{y}_i, \mathbf{h}_i|\mathbf{x}_i, n_i)$ using a log-linear model:

$$p(\mathbf{y}_i, \mathbf{h}_i|\mathbf{x}_i, n_i) = \frac{\exp \mathbf{w}^T \mathbf{\Phi}(\mathbf{x}_i, \mathbf{y}_i, \mathbf{h}_i)}{Z(\mathbf{x}_i, n_i)}, \quad (3)$$

where $Z(\mathbf{x}_i, n_i) = \sum_{\mathbf{y}} \sum_{\mathbf{h}} \exp \mathbf{w}^T \mathbf{\Phi}(\mathbf{x}_i, \mathbf{y}, \mathbf{h})$. To keep our models tractable, we assume our feature function $\mathbf{\Phi}$ decomposes linearly, similar to a linear-chain graphical model:

$$\mathbf{\Phi}(\mathbf{x}_i, \mathbf{y}_i, \mathbf{h}_i) = \sum_{n=1}^{n_i} \phi(X_{i,m}, Y_{i,n}, m, n, m'),$$

where $\mathbf{h}_i[n] = m$ and $\mathbf{h}_i[n-1] = m'$. Therefore, each factor in our linear chain graph structure depends on the alignment state for the current and the previous video chunk. For any two consecutive alignment states $\mathbf{h}_i[n] = m$ and $\mathbf{h}_i[n-1] = m'$, we represent the factor potential as:

$$\Psi(X_{i,m}, Y_{i,n}, m, n, m') = \exp\left[\mathbf{w}^T \phi(X_{i,m}, Y_{i,n}, m, n, m')\right]$$

Our goal is to maximize the following log-likelihood function:

$$L(\mathbf{w}) = \sum_{i=1}^{N} \log \sum_{\mathbf{h}_i} p(\mathbf{y}_i, \mathbf{h}_i|\mathbf{x}_i, n_i). \quad (4)$$

The gradient of the log-likelihood function with respect to the weight parameters is:

$$\frac{\partial L}{\partial \mathbf{w}} = \sum_{i=1}^{N} \Big[\mathbb{E}_{p(\mathbf{h}|\mathbf{x}_i, n_i, \mathbf{y}_i)} \left[\mathbf{\Phi}(\mathbf{x}_i, \mathbf{y}_i, \mathbf{h})\right] - \mathbb{E}_{p(\mathbf{y}, \mathbf{h}|\mathbf{x}_i, n_i)} \left[\mathbf{\Phi}(\mathbf{x}_i, \mathbf{y}, \mathbf{h})\right] \Big] \quad (5)$$

We apply the stochastic gradient descent algorithm (Vishwanathan et al., 2006) to maximize the conditional log-likelihood. For each observation $(\mathbf{x}_i, \mathbf{y}_i)$, we perform forward-backward dynamic programming to estimate the two expectation terms in equation 5, as discussed next.

4.1.1 Estimation of $\mathbb{E}_{p(\mathbf{h}|\mathbf{x}_i, n_i, \mathbf{y}_i)} \left[\mathbf{\Phi}(\mathbf{x}_i, \mathbf{y}_i, \mathbf{h})\right]$

To estimate the first expectation term in equation 5, we need to sum over all the possible alignment states $h[n] = m$, where $n \in \{1, \ldots, n_i\}$ and $m \in \{1, \ldots, m_i\}$. Since the output sequence $\mathbf{y}_i$ is given, we refer to this stage as "forced" forward-backward stage. The forward messages $\alpha_n^F[m] \propto p(Y_{i,1}, \ldots, Y_{i,n}, \mathbf{h}[n] = m \mid \mathbf{x}_i)$ are estimated using the following recursion:

$$\alpha_n^F(m) = \sum_{m'} \alpha_{n-1}^F(m')\ \Psi(X_{i,m}, Y_{i,n}, m, n, m')$$

where m' is one of the predecessors of the alignment state $h[n] = m$. Assuming no restrictions on the possible alignments, the computational complexity of each iteration on a single observation pair $(\mathbf{x}_i, \mathbf{y}_i)$ is $O(m_i^2 n_i d)$ for m_i sentences, n_i video chunks, and d dimensional features. However, we allow only a constant number of predecessor and successor states for each alignment state, and hence the computational complexity becomes $O(m_i n_i d)$. Similarly, we apply backward recursions, with the same computational complexity.

4.1.2 Estimation of $\mathbb{E}_{p(\mathbf{y}, \mathbf{h}|\mathbf{x}_i, n_i)} \left[\mathbf{\Phi}(\mathbf{x}_i, \mathbf{y}, \mathbf{h})\right]$

While computing the second expectation term, we assume only $\mathbf{x}_i$ and the number of video chunks n_i are observed, and we need to sum probabilities over all possible alignments $h[n] = m$ and all possible video sequences $\mathbf{y}$. Again we apply forward-backward. The computational complexity, however, grows significantly, as we need to sum over all possible set of blobs that may be touched by hands in

each video segment. The forward message $\alpha_n(m)$ is computed as:

$$\alpha_n(m) = \sum_{m'} \alpha_{n-1}(m') \sum_{Y \subseteq V_Y} \Psi(X_{im}, Y, m, n, m')$$

There can be $2^{|V_Y|} - 1$ possible subset of blobs at each of the alignment position, and the overall computational complexity becomes $O(2^{|V_Y|} m_i n_i d)$, which is prohibitively expensive, even for a small number of blobs. In our videos, the hands never touch more than 3 objects at a time. So we considered only the non-empty subsets with 3 or less elements: $P = \{S : S \subseteq V_Y, |S| \leq 3, S \neq \emptyset\}$. The pruning of larger subsets reduces the complexity to $O(|V_Y|^3 m_i n_i d)$. We can further reduce computation by decomposing the forward-backward recursions to the co-occurrence features and alignment path features:

$$\Psi(X_{im}, Y, m, n, m') = \Psi_{co}(X_{im}, Y) \Psi_{ap}(m, n, m')$$

The potential due to alignment path features (Ψ_{ap}) does not depend on the subset of blobs, and only depends on the current and previous alignment states $h[n] = m$ and $h[n-1] = m'$. On the other hand, the co-occurrence potential Ψ_{co} for a given set of blobs Y depends only on the sentence that it is being aligned to, and does not depend on the video chunk index n. Therefore we can decompose the forward recursion as:

$$\alpha_n(m) = \sum_{m'} \alpha_{n-1}(m') \; \Psi_{ap}(m, n, m') \; \delta(m)$$

where $\delta(m) = \sum_{Y \in P} \Psi_{co}(X_{im}, Y)$. We can precompute the values of $\delta(m)$ for each of the m_i sentences, which takes $O(m_i d |V_Y|^3)$ operations. Finally, we run forward recursions over all the alignment states using the precomputed values, and the complexity becomes $O(m_i d |V_Y|^3 + m_i n_i d)$ Similarly the backward recursion becomes:

$$\beta_n(m) = \sum_{m'} \beta_{n+1}(m') \; \Psi_{ap}(m', n{+}1, m) \; \delta(m')$$

The alignment state transition probabilities $\xi_n(m', m)$ represents the probability $p(\mathbf{h}_{n-1} = m', \mathbf{h}_n = m \mid \mathbf{x}_i)$, which can be estimated by marginalizing over all possible sets of blobs:

$$\xi_n(m', m) \propto \alpha_{n-1}(m') \Psi_{ap}(m, n, m') \delta(m) \beta_n(m)$$

4.2 Latent Variable Structured Perceptron

Structured Perceptron (Collins, 2002) has become a popular method for discriminative structured learning due to its relatively fast convergence rate and theoretical convergence guarantee. Since true alignments are unknown, we apply the latent variable structured perceptron algorithm (Liang et al., 2006; Sun et al., 2009; Yu et al., 2013) for our discriminative alignment task.

We iteratively scan through our dataset, one protocol and video pair $(\mathbf{x}_i, \mathbf{y}_i)$ at a time. First, we infer the best alignment $\mathbf{h}_i^{Forced}$ for the given observation pair $(\mathbf{x}_i, \mathbf{y}_i)$ and the current weight vector $\mathbf{w}$:

$$\mathbf{h}_i^{Forced} = \arg\max_{\mathbf{h}} \mathbf{w}^T \mathbf{\Phi}(\mathbf{x}_i, \mathbf{y}_i, \mathbf{h}). \qquad (6)$$

This step is known as Forced Decoding, as we are given both the protocol sentences and the associated video chunks. Forced decoding is performed using Viterbi-like dynamic programming (Algorithm 1), where the dynamic programming states are the alignment states (m, n) such that $h[n] = m$.

Algorithm 1 Perceptron Forced-Decoding

Input: Observation pair $(\mathbf{x}_i, \mathbf{y}_i)$ and a weight vector $\mathbf{w}$.
$m_i \leftarrow$ length($\mathbf{x}_i$), and $n_i \leftarrow$ length($\mathbf{y}_i$),
$D[m, n] \leftarrow -\infty$ for $0 \leq m \leq m_i$ and $0 \leq n \leq n_i$
$D[0, 0] \leftarrow 0$
for $m = 1$ to m_i **do**
  for $n = 1$ to n_i **do**
    for each $(m', n-1) \in$ Predecessors(m, n) **do**
      $\mathbf{\Phi} \leftarrow$ create-features$(X_{i,m}, Y_{i,n}, m, n, m')$
      if $D[m', n-1] + \mathbf{w}^T \mathbf{\Phi} > D[m, n]$ **then**
        $D[m, n] \leftarrow D[m', n-1] + \mathbf{w}^T \mathbf{\Phi}$
        Backpointers$[m, n] \leftarrow m'$
$\mathbf{h}_i^{Forced} \leftarrow$ Backtrack(D, Backpointers)
Return $\mathbf{h}_i^{Forced}$

Next, we decode both the highest scoring alignment $\hat{\mathbf{h}}_i$ and video sequence $\hat{\mathbf{y}}_i$, given the protocol $\mathbf{x}_i$ and the number of video chunks n_i.

$$\hat{\mathbf{h}}_i, \hat{\mathbf{y}}_i = \arg\max_{\mathbf{h}, \mathbf{y}} \mathbf{w}^T \mathbf{\Phi}(\mathbf{x}_i, \mathbf{y}, \mathbf{h}) \qquad (7)$$

We refer to this step as Full Decoding (Algorithm 2). The dynamic programming is similar to that for forced decoding, except that we need to find the best set of blobs given a set of nouns, for every protocol sentence $X_{i,m}$:

$$B[m] = \arg\max_{S \in P} \mathbf{w}_{co}^T \mathbf{\Phi_{co}}(X_{i,m}, S) \qquad (8)$$

where P is the pruned set of blobs and $\mathbf{\Phi_{co}}(X_{i,m}, S)$ is a vector containing only the co-occurrence features, and $\mathbf{w}_{co}$ contains their corresponding weights. The detailed algorithm is described in Algorithm 2. Finally, we update the weight vector $\mathbf{w}$:

$$\mathbf{w}^{(new)} = \mathbf{w}^{(old)} + \mathbf{\Phi}(\mathbf{x}_i, \mathbf{y}_i, \mathbf{h}_i^{Forced}) - \mathbf{\Phi}(\mathbf{x}_i, \hat{\mathbf{y}}_i, \hat{\mathbf{h}}_i)$$

Algorithm 2 Perceptron Full Decoding

Input: Input protocol $\mathbf{x}_i$, set of all blobs V_Y, number of video chunks n_i, weight vector $\mathbf{w}$.

$m_i \leftarrow \text{length}(\mathbf{x}_i)$
$D[m, n] \leftarrow -\infty$ for $0 \leq m \leq m_i$ and $0 \leq n \leq n_i$
$B[m] \leftarrow \emptyset$ for $0 \leq m \leq m_i$
$D[0, 0] \leftarrow 0$
$P \leftarrow \{S : S \subset V_Y, |S| \leq 3, S \neq \emptyset\}$ // precompute the pruned list of subsets of blobs
for $m = 1$ to m_i **do**
  $B[m] \leftarrow \arg\max_{S \in P} \mathbf{w}_{co}^T \mathbf{\Phi_{co}}(X_{i,m}, S)$
  for $n = 1$ to n_i **do**
    for each $(m', n-1) \in$ Predecessors(m, n) **do**
      $\mathbf{\Phi} \leftarrow$ create-features$(X_{i,m}, B[m], m, n, m')$
      if $D[m', n-1] + \mathbf{w}^T \mathbf{\Phi} > D[m, n]$ **then**
        $D[m, n] \leftarrow D[m', n-1] + \mathbf{w}^T \mathbf{\Phi}$
        Backpointer$[m, n] \leftarrow m'$
$\hat{\mathbf{h}}_i \leftarrow$ Backtrack$(D$, Backpointers$)$
$\hat{\mathbf{y}}_i \leftarrow [B[\hat{\mathbf{h}}_{i,1}], \ldots, B[\hat{\mathbf{h}}_{i,n_i}]]$
Return $\hat{\mathbf{h}}_i, \hat{\mathbf{y}}_i$

4.3 Constrained Decoding

During the full decoding of $(\hat{\mathbf{h}}_i, \hat{\mathbf{y}}_i)$, we have no information regarding how many video chunks to assign to each sentence. As a result, the full decoding is unlikely to predict the correct video sequence, no matter how many training iterations performed. In practice, the unconstrained full decoding often ends up aligning too many video chunks to one of the protocol sentences.

To address this problem, we modified the perceptron update rule. Instead of performing unconstrained full decoding, we constrain the alignment $\hat{\mathbf{h}}_i$ to be same as the forced alignment $\mathbf{h}_i^{Forced}$, and infer the best sequence of video chunks $\hat{\mathbf{y}}_i^{Constr}$ under this constraint:

$$\hat{\mathbf{y}}_i^{Constr} = \arg\max_{\mathbf{y}} \mathbf{w}^T \mathbf{\Phi}(\mathbf{x}_i, \mathbf{y}, \mathbf{h}_i^{Forced})$$

We refer to this decoding step as "constrained decoding" (Algorithm 3), and refer to this constrained LSP variant as LSP-C. The modified weight update rule is:

$$\mathbf{w}^{(new)} = \mathbf{w}^{(old)} + \mathbf{\Phi}(\mathbf{x}_i, \mathbf{y}_i, \mathbf{h}_i^{Forced}) - \mathbf{\Phi}(\mathbf{x}_i, \hat{\mathbf{y}}_i^{Constr}, \mathbf{h}_i^{Forced})$$

Algorithm 3 Perceptron Constrained-Decoding

Input: Input protocol $\mathbf{x}_i$, set of all blobs V_Y, number of video chunks n_i, forced alignment $\mathbf{h}_i^{Forced}$, weight vector $\mathbf{w}$.

$P \leftarrow \{S : S \subset V_Y, |S| \leq 3, S \neq \emptyset\}$
for $n = 1$ to n_i **do**
  $m \leftarrow \mathbf{h}_i^{Forced}[n]$
  $\hat{Y}_{i,n}^{Constr} \leftarrow \arg\max_{S \in P} \mathbf{w}_{co}^T \mathbf{\Phi_{co}}(X_{i,m}, S)$
Return $\hat{\mathbf{y}}_i^{Constr} = [\hat{Y}_{i,1}^{Constr}, \ldots, \hat{Y}_{i,n_i}^{Constr}]$

4.4 Latent Structured SVM

Structured SVM can be formulated by extending structured perceptron with two simple modifications: (1) incorporating a large-margin regularization term, and (2) incorporating a general loss function, instead of the zero-one loss of perceptron. The regularization reduces overfitting by keeping feature weights relatively small. Let the loss-augmented full decoding be:

$$(\hat{\mathbf{y}}_i, \hat{\mathbf{h}}_i) = \arg\max_{\mathbf{y}, \mathbf{h}} \mathbf{w}^T \mathbf{\Phi}(\mathbf{x}_i, \mathbf{y}, \mathbf{h}) + \mathcal{L}_i(\mathbf{y}, \mathbf{h}),$$

where $\mathcal{L}_i(\mathbf{y}, \mathbf{h})$ is the loss function for the i^{th} observation. LSSVM minimizes the following objective function:

$$C(w) = \frac{1}{N} \sum_{i=1}^{N} \Big(\mathbf{w}^T \mathbf{\Phi}(\mathbf{x}_i, \hat{\mathbf{y}}_i, \hat{\mathbf{h}}_i) + \mathcal{L}_i(\hat{\mathbf{y}}_i, \hat{\mathbf{h}}_i) - \mathbf{w}^T \mathbf{\Phi}(\mathbf{x}_i, \mathbf{y}_i, \mathbf{h}_i^{Forced}) \Big) + \frac{\lambda}{2} \|w\|^2,$$

which is non-convex and non-differentiable, and optimized utilizing the subgradient method (Ratliff et al., 2007). We perform online learning, and the subgradient in each iteration is:

$$g_i(\mathbf{w}) = \mathbf{\Phi}(\mathbf{x}_i, \hat{\mathbf{y}}_i, \hat{\mathbf{h}}_i) - \mathbf{\Phi}(\mathbf{x}_i, \mathbf{y}_i, \mathbf{h}_i^{Forced}) + \lambda \mathbf{w}.$$

Similar to LSP-C, we can obtain a constrained variant LSSVM-C, by replacing loss-augmented decoding with a constrained variant, where we fix $\hat{\mathbf{h}}_i$ to forced alignment $\mathbf{h}_i^{Forced}$.

4.5 Latent Variables to Map Blobs to Nouns

Given a sentence $X_{i,n}$ and a video segment $Y_{i,m}$, we further introduce additional latent variables to map each blob in $Y_{i,m}$ to one of the nouns in $X_{i,n}$. These latent variables are similar to the IBM Model 1 latent variables of Naim et al. (2014). Instead of turning on the (noun, blob) co-occurrence feature for every noun and blob in $X_{i,n}$ and $Y_{i,m}$, the latent variables map each blob to one of the nouns only. For LCRF, we sum over all the latent variables for estimating the expectations. For LSP and LSSVM, the (noun,blob) feature with maximum feature weight triggers for each blob.

5 Feature Design

The features used in our discriminative models can be grouped in two categories: (1) co-occurrence features, and (2) alignment path features. The co-occurrence features depend only on a protocol sentence and the video segment it aligns to. The alignment path features, on the other hand, do not depend on the co-occurrence of sentence and video segment, and instead capture general alignment properties, e.g., jump size and the distance of an alignment state from the diagonal.

5.1 Co-occurrence Features

The co-occurrence features included in our experiments are:

- *Co-occurrence of Nouns and Blobs:* For each noun in the input protocols and each blob in the videos, we add a boolean feature (noun, blob), which is turned on if we align a sentence containing that noun with a video segment containing that blob.

- *Co-occurrence of Verbs and Blobs:* For each verb in the input protocols and each blob in the videos, we add a boolean feature. This feature captures the observation that certain verbs are more likely to occur with certain objects (e.g., 'write' co-occurs with 'pen', 'aspirate' co-occurs with 'pipette').

We experimented with co-occurrence features of the form: (noun, verb, blob) triplets. However, including these features did not provide any noticeable gain, while significantly increasing the computation time, as the number of features increased drastically. Therefore, we did not include these features in our final experiments.

5.2 Alignment Path Features

Alignment path features depend on the current alignment state $\mathbf{h}[n] = m$, and the previous alignment states $\mathbf{h}[n-1] = m'$. These features do not depend on the nouns and verbs in the sentences and the blobs in the video segments. We used the following alignment path features:

- *Jump Size:* Since we allow monotonic jumps only, the jump sizes can be either zero or one. Therefore, we added two features for these two jump sizes.

- *Positional Features:* we added positional features (Dyer et al., 2011) to discourage alignment states that are too far from the diagonal. For each alignment state (m, n), we estimate normalized distance from the diagonal as $|\frac{m}{m_i} - \frac{n}{n_i}|$. Again we used boolean features by assigning this normalized distance to five equally spaced bins.

The alignment features are not updated by the LSP-C and LSSVM-C methods, as they assume $\mathbf{h}_i^{Forced}$ and $\hat{\mathbf{h}}_i$ to be identical.

6 Results

Our dataset contains 12 wetlab experiment videos, for 3 different protocols (4 videos per protocol). Each protocol contains natural language instructions for an actual biological experiment. On average, each protocol has 9 steps, and 24 sentences. The videos are recorded using an RGB-D Kinect camera, in a mock wetlab setup. The average video length is $\sim$ 5 minutes. There are 34 unique nouns and 25 unique verbs in the protocols, and 22 distinct blobs in the videos.

We follow the same data pre-processing technique as described by Naim et al. (2014). The number of blobs is assumed to be known apriori. We oversegment each frame into many superpixels using the SLIC Superpixels algorithm (Achanta et al., 2012). We combine multiple adjacent superpixels into a blob, based on a pre-trained Gaussian mixture

Dataset	Average Alignment Accuracy (%)							
	LHMM	LCRF	LSP	LSP-C	LSP-H	LSSVM	LSSVM-C	LSSVM-H
Manual-Tracking	75.58	**85.09**	79.64	80.68	80.41	79.64	80.68	80.41
Auto-Tracking	64.04	**65.59**	61.99	63.95	65.27	61.99	63.95	65.27

Table 1: Alignment accuracy (% of video chunks aligned to the correct protocol step) for both manual and automatic tracking data. LHMM is the existing state-of-the-art generative model. For the variants of latent perceptron (LSP) and latent structured SVM (LSSVM), "C" indicates constrained decoding, and "H" indicates hybrid update.

color model and their boundary maps (Luo and Guo, 2003), and track each blob using a 3D Kalman filter. In order to isolate alignment error from computer vision tracking and segmentation error, we manually tracked and annotated each of the video segments with the set of blobs touched by hands using the video annotation tool Anvil (Kipp, 2012). The alignment accuracies are reported both for the manual and automated tracking datasets. Parsing error is relatively small. The Charniak-Johnson parser correctly identified the nouns and verbs for most sentences, except for several single-word imperative sentences (e.g., Mix.), for which the verbs were mistakenly parsed as nouns.

We experimented with the latent CRF (LCRF), latent perceptron (LSP) and its constrained variant (LSP-C), and latent SVM (LSSVM) and its constrained variant (LSSVM-C). Furthermore, we tried two hybrid variants LSP-H and LSSVM-H, where we started with constrained decoding, and later switched to full decoding. We experimented by incorporating additional latent variables for Blob-to-Noun mapping (Section 4.5), which significantly improved alignment accuracy for LCRF, but decreased accuracy for LSP and LSSVM and their variants. We report the best result for each model. The discriminative algorithms are compared with the state-of-the-art LHMM model (Naim et al., 2014), which is a generative HMM with latent variables for blob-to-noun mapping and the observation states of each noun.

We initialized the weights for co-occurrence and jump size features to the log-probabilities learned by the generative HMM model. All the other features are initialized to zero. For both LHMM and the discriminative models, we used monotonic jumps as they performed better than the non-monotonic jumps. We used the same learning rate $\eta = \frac{0.001}{\sqrt{t}}$ (where t is the iteration number) for all the discriminative models, and the LSSVM regularization constant $\lambda = 0.001$. All the Perceprton and SVM variants performed "weight averaging" (Collins, 2002). The number of iterations are set to 100 for all the algorithms.

Table 1 shows that the discriminative models, especially LCRF and LSP-H/LSSVM-H, outperform the generative model LHMM both on the manual-tracking and auto-tracking datasets. For the manual-tracking dataset, the difference between LHMM and each of the discriminative models is statistically significant (p-value < 0.0001). On the auto-tracking dataset, however, the differences are not significant (p-value > 0.1). Table 2 shows an example of an alignment obtained by LCRF for a short segment of a manually tracked video.

The average running time for each iteration per video is 0.8 seconds for LHMM, 1.1 seconds for LSP and LSSVM, and 2.5 seconds for LCRF on a 2.9 GHz Intel Core-i7 processor and 8GB RAM.

7 Discussions and Future Work

The results show that discriminative methods outperform the generative LHMM model on both the manual and auto-tracking datasets. We achieved the best overall accuracy using the LCRF model. LCRF takes expectations over all possible alignment states and video sequences. On the other hand, LSP and LSSVM consider the highest scoring prediction only, which is similar to the hard-decision decoding. With no information regarding how many video segments to align to each sentence, LSP and LSSVM could not correctly predict the output video sequences during full decoding, and the weight vectors did not converge. By constraining the alignment to the forced alignment, we avoid aggressive updates, which may have helped LSP-C and LSSVM-C to learn better alignments. However, constrained decoding has a limitation that it can not update align-

Start (s)	End (s)	Blobs in Hands	Detected Nouns	Detected Verbs	Protocol Sentence
40.58	42.58	boat	boat, scale	place	place the plastic boat on the scale .
42.58	42.90	boat	scale		zero the scale .
42.90	48.48	base	spatula, base, boat	measure	using the spatula , measure 20 g of lb broth base into the plastic boat .
48.48	58.95	base, spatula	spatula, base, boat	measure	using the spatula , measure 20 g of lb broth base into the plastic boat .
58.95	65.93	base	spatula, base, boat	measure	using the spatula , measure 20 g of lb broth base into the plastic boat .
65.93	80.90	boat, bottle	base, bottle	pour	pour the lb broth base into the 1000 ml bottle .
83.80	84.80	water	water	add	add 800 ml of di water .
84.80	88.95	water	water, sink	use	use the di water near the sink .
88.95	96.68	water, bottle	water, sink	use	use the di water near the sink .
96.68	104.67	water	mix		mix .
108.15	118.12	bottle	cap, bottle, water	put, shake, mix	put a cap on the bottle and shake to mix the dry ingredients with the water .

Table 2: An example of an alignment, obtained for a part of a manually tracked video. We notice several incorrect parses, e.g., the verbs "mix" and "zero" were not detected correctly.

ment path features. LCRF sums over all possible output and latent variables, which includes the correct solution, and hence constrained decoding is not necessary. While the latent variables for blob-to-noun mappings improved the alignment accuracy for LCRF, it did not improve alignment accuracy for LSP and LSSVM and their variants, presumably because of their hard-decision decoding approach.

Among the different variants of LSP and LSSVM, we obtained the best accuracy with the hybrid variants (LSP-H and LSSVM-H), where we started with constrained decoding, and then switched to standard updates. While these hybird approaches provided better accuracy, they still suffer from the issue of not converging. The feature weights learned by LSSVM and its variants were smaller than that for LSP (due to regularization). However, they always resulted in the same forced decoding alignments in our experiments, and obtained same alignment accuracy.

Unlike the previous models, we considered the co-occurrences of verbs with blobs in the video. The highest weighted features include: *(write, pen)*, *(aspirate, pipette)*, which agree with our intuition. Our immediate next step will be to automatically learn a dictionary of hand motion patterns, and consider the co-occurrence of these patterns with verbs in the sentences. Some of the objects in our video are small and thin (e.g., pen, pipette, spatula, plastic boat), and were not reliably detected by the computer vision segmentation and tracking system. This may be the reason why we achieved relatively smaller improvements on the auto-tracking dataset.

Our alignment models are different from the traditional discriminative approaches in that our cost function is not same as our evaluation criteria. Although our goal is to improve alignment accuracy, the objective function that we minimize is either the negative conditional log-likelihood (LCRF) or the number of mis-predicted video segments (LSSVM). Since the ground truth alignments are unknown, we could not integrate alignment error in our objective function. The proposed discriminative models outperform LHMM despite the fact that the discriminative models are simpler – lacking latent variables for the observation states of nouns. The alignment accuracy of the discriminative models is expected to improve even further once these latent variables are incorporated.

8 Conclusion

We proposed three discriminative unsupervised alignment algorithms and their novel variants using constrained decoding. The proposed algorithms incorporate overlapping features to capture the co-occurrences of nouns and verbs with video blobs, and outperform the state-of-the-art latent HMM model via discriminative training.

Acknowledgments Funded by NSF IIS-1446996, ONR N00014-11-10417, Intel ISTCPC, DoD SBIR N00014-12-C-0263, DARPA FA8750-13-2-0041 (DEFT), NSF IIS-1449278, and a Google Faculty Research Award.

References

R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and Sabine Susstrunk. 2012. SLIC superpixels compared to state-of-the-art superpixel methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(11):2274–2282.

C. Glenn Begley and Lee M. Ellis. 2012. Drug development: Raise standards for preclinical cancer research. *Nature*, 483(7391):531–533.

Phil Blunsom and Trevor Cohn. 2006. Discriminative word alignment with conditional random fields. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics*, ACL-44, pages 65–72, Stroudsburg, PA, USA. Association for Computational Linguistics.

Peter F. Brown, Stephen A. Della Pietra, Vincent J. Della Pietra, and Robert L. Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19(2):263–311.

Eugene Charniak and Mark Johnson. 2005. Coarse-to-fine n-best parsing and maxent discriminative reranking. In *ACL*.

Michael Collins. 2002. Discriminative training methods for Hidden Markov Models: Theory and experiments with perceptron algorithms. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing*.

A. P. Dempster, N. M. Laird, and D. B. Rubin. 1977. Maximum likelihood from incomplete data via the *EM* algorithm. *Journal of the Royal Statistical Society*, 39(1):1–21.

Chris Dyer, Jonathan Clark, Alon Lavie, and Noah A Smith. 2011. Unsupervised word alignment with arbitrary features. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 409–419. Association for Computational Linguistics.

Rohit J Kate and Raymond J Mooney. 2007. Learning language semantics from ambiguous supervision. In *AAAI*, volume 7, pages 895–900.

M Kipp. 2012. Anvil: A universal video research tool. *Handbook of Corpus Phonology. Oxford University Press*.

Thomas Kollar, Stefanie Tellex, Deb Roy, and Nicholas Roy. 2010. Toward understanding natural language directions. In *Human-Robot Interaction (HRI), 2010 5th ACM/IEEE International Conference on*, pages 259–266. IEEE.

Niveda Krishnamoorthy, Girish Malkarnenkar, Raymond Mooney, Kate Saenko, and Sergio Guadarrama. 2013. Generating natural-language video descriptions using text-mined knowledge. In *Proceedings of the National Conference on Artificial Intelligence (AAAI-13)*, volume 2013, page 3.

Jayant Krishnamurthy and Thomas Kollar. 2013. Jointly learning to parse and perceive: Connecting natural language to the physical world. *Transactions of the Assoc. for Comp. Ling.*, 10:193–206.

Percy Liang, Alexandre Bouchard-Côté, Dan Klein, and Ben Taskar. 2006. An end-to-end discriminative approach to machine translation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 761–768, Sydney, Australia, July.

Jiebo Luo and Cheng-en Guo. 2003. Perceptual grouping of segmented regions in color images. *Pattern Recognition*, 36(12):2781 – 2792.

Cynthia Matuszek, Nicholas Fitzgerald, Luke Zettlemoyer, Liefeng Bo, and Dieter Fox. 2012. A joint model of language and perception for grounded attribute learning. In *Proceedings of the 29th International Conference on Machine Learning (ICML-2012)*, pages 1671–1678.

Iftekhar Naim, Young Song, Qiguang Liu, Henry Kautz, Jiebo Luo, and Daniel Gildea. 2014. Unsupervised alignment of natural language instructions with video segments. In *Proceedings of Twenty-Eighth AAAI Conference on Artificial Intelligence*.

Lawrence R. Rabiner. 1989. A tutorial on Hidden Markov Models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286.

Nathan Ratliff, J. Andrew (Drew) Bagnell, and Martin Zinkevich. 2007. (Online) subgradient methods for structured prediction. In *Eleventh International Conference on Artificial Intelligence and Statistics (AISTATS)*, March.

M. Rohrbach, Wei Qiu, I. Titov, S. Thater, M. Pinkal, and B. Schiele. 2013. Translating video content to natural language descriptions. In *14th IEEE International Conference on Computer Vision (ICCV)*, pages 433–440, Dec.

X. Sun, T. Matsuzaki, D. Okanohara, and J. Tsujii. 2009. Latent variable perceptron algorithm for structured classification. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence (IJCAI 2009)*, pages 1236–1242.

Ben Taskar, Simon Lacoste-Julien, and Dan Klein. 2005. A discriminative matching approach to word alignment. In *Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing*, HLT '05, pages 73–80, Stroudsburg, PA, USA. Association for Computational Linguistics.

Stefanie Tellex, Pratiksha Thaker, Joshua Joseph, and Nicholas Roy. 2013. Learning perceptually grounded

word meanings from unaligned parallel data. *Machine Learning*, pages 1–17.

SVN Vishwanathan, Nicol N Schraudolph, Mark W Schmidt, and Kevin P Murphy. 2006. Accelerated training of conditional random fields with stochastic gradient methods. In *Proceedings of the 23rd international conference on Machine learning*, pages 969–976. ACM.

Stephan Vogel, Hermann Ney, and Christoph Tillmann. 1996. HMM-based word alignment in statistical translation. In *COLING-96*, pages 836–841, Copenhagen, Denmark.

Chen Yu and Dana H Ballard. 2004. On the integration of grounding language and learning objects. In *AAAI*, volume 4, pages 488–493.

Haonan Yu and Jeffrey Mark Siskind. 2013. Grounded language learning from video described with sentences. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (ACL-13)*, volume 1, pages 53–63.

Heng Yu, Liang Huang, Haitao Mi, and Kai Zhao. 2013. Max-violation perceptron and forced decoding for scalable MT training. In *Conference on Empirical Methods in Natural Language Processing (EMNLP-13)*.