

# Event Representation with Sequential, Semi-Supervised Discrete Variables

Mehdi Rezaee

Department of Computer Science  
University of Maryland Baltimore County  
Baltimore, MD 21250 USA  
rezaee1@umbc.edu

Francis Ferraro

Department of Computer Science  
University of Maryland Baltimore County  
Baltimore, MD 21250 USA  
ferraro@umbc.edu

## Abstract

Within the context of event modeling and understanding, we propose a new method for neural sequence modeling that takes partially-observed sequences of discrete, external knowledge into account. We construct a sequential neural variational autoencoder, which uses Gumbel-Softmax reparametrization within a carefully defined encoder, to allow for successful backpropagation during training. The core idea is to allow semi-supervised external discrete knowledge to *guide*, but not restrict, the variational latent parameters during training. Our experiments indicate that our approach not only outperforms multiple baselines and the state-of-the-art in narrative script induction, but also converges more quickly.

## 1 Introduction

Event scripts are a classic way of summarizing events, participants, and other relevant information as a way of analyzing complex situations (Schank and Abelson, 1977). To learn these scripts we must be able to group similar-events together, learn common patterns/sequences of events, and learn to represent an event’s arguments (Minsky, 1974). While continuous embeddings can be learned for events and their arguments (Ferraro et al., 2017; Weber et al., 2018a), the direct inclusion of more structured, discrete knowledge is helpful in learning event representations (Ferraro and Van Durme, 2016). Obtaining fully accurate structured knowledge can be difficult, so when the external knowledge is neither sufficiently reliable nor present, a natural question arises: how can our models use the knowledge that *is* present?

Generative probabilistic models provide a framework for doing so: external knowledge is a random variable, which can be observed or latent, and the data/observations are generated (explained) from it. Knowledge that is discrete, sequential, or both—

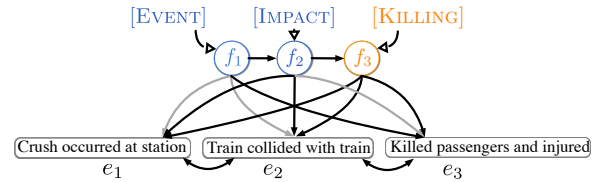


Figure 1: An overview of event modeling, where the observed events (black text) are generated via a sequence of semi-observed random variables. In this case, the random variables are directed to take on the meaning of semantic frames that can be helpful to explain the events. Unobserved frames are in orange and observed frames are in blue. Some connections are more important than others, indicated by the weighted arrows.

such as for script learning—complicates the development of neural generative models.

In this paper, we provide a successful approach for incorporating partially-observed, discrete, sequential external knowledge in a neural generative model. We specifically examine *event sequence modeling* augmented by *semantic frames*. Frames (Minsky, 1974, i.a.) are a semantic representation designed to capture the common and general knowledge we have about events, situations, and things. They have been effective in providing crucial information for modeling and understanding the meaning of events (Peng and Roth, 2016; Ferraro et al., 2017; Padia et al., 2018; Zhang et al., 2020). Though we focus on semantic frames as our source of external knowledge, we argue this work is applicable to other similar types of knowledge.

We examine the problem of modeling observed *event tuples* as a partially observed sequence of **semantic frames**. Consider the following three events, preceded by their corresponding bracketed frames, from Fig. 1:

[EVENT] *crash occurred at station.*

[IMPACT] *train collided with train.*

[KILLING] *killed passengers and injured.*

We can see that even without knowing the [KILLING] frame, the [EVENT] and [IMPACT]

frames can help predict the word *killed* in the third event; the frames summarize the events and can be used as guidance for the latent variables to represent the data. On the other hand, words like *crash*, *station* and *killed* from the first and third events play a role in predicting [IMPACT] in the second event. Overall, to successfully represent events, beyond capturing the event to event connections, we propose to consider all the information from the frames to events and frames to frames.

In this work, we study the effect of tying discrete, sequential latent variables to partially-observable, noisy (imperfect) semantic frames. Like [Weber et al. \(2018b\)](#), our semi-supervised model is a bidirectional auto-encoder, with a structured collection of latent variables separating the encoder and decoder, and attention mechanisms on both the encoder and decoder. Rather than applying vector quantization, we adopt a Gumbel-Softmax ([Jang et al., 2017](#)) ancestral sampling method to easily switch between the observed frames and latent ones, where we inject the observed frame information on the Gumbel-Softmax parameters before sampling. Overall, our contributions are:

- We demonstrate how to learn a VAE that contains sequential, discrete, and *partially-observed* latent variables.
- We show that adding partially-observed, external, semantic frame knowledge to our structured, neural generative model leads to improvements over the current state-of-the-art on recent core event modeling tasks. Our approach leads to faster training convergence.
- We show that our semi-supervised model, though developed as a generative model, can effectively predict the labels that it may not observe. Additionally, we find that our model outperforms a discriminatively trained model with full supervision.

## 2 Related Work

Our work builds on both event modeling and latent generative models. In this section, we outline relevant background and related work.

### 2.1 Latent Generative Modeling

Generative latent variable models learn a mapping from the low-dimensional hidden variables  $\mathbf{f}$  to the observed data points  $\mathbf{x}$ , where the hidden representation captures the high-level information to

explain the data. Mathematically, the joint probability  $p(\mathbf{x}, \mathbf{f}; \theta)$  factorizes as follows

$$p(\mathbf{x}, \mathbf{f}; \theta) = p(\mathbf{f})p(\mathbf{x}|\mathbf{f}; \theta), \quad (1)$$

where  $\theta$  represents the model parameters. Since in practice maximizing the log-likelihood is intractable, we approximate the posterior by defining  $q(\mathbf{f}|\mathbf{x}; \phi)$  and maximize the ELBO ([Kingma and Welling, 2013](#)) as a surrogate objective:

$$\mathcal{L}_{\theta, \phi} = \mathbb{E}_{q(\mathbf{f}|\mathbf{x}; \phi)} \log \frac{p(\mathbf{x}, \mathbf{f}; \theta)}{q(\mathbf{f}|\mathbf{x}; \phi)}. \quad (2)$$

In this paper, we are interested in studying a specific case; the input  $\mathbf{x}$  is a sequence of  $T$  tokens, we have  $M$  sequential discrete latent variables  $\mathbf{z} = \{z_m\}_{m=1}^M$ , where each  $z_m$  takes  $F$  discrete values. While there have been effective proposals for unsupervised optimization of  $\theta$  and  $\phi$ , we focus on learning *partially observed* sequences of these variables. That is, we assume that in the training phase some values are observed while others are latent. We incorporate this partially observed, external knowledge to the  $\phi$  parameters to guide the inference. The inferred latent variables later will be used to reconstruct to the tokens.

[Kingma et al. \(2014\)](#) generalized VAEs to the semi-supervised setup, but they assume that the dataset can be split into observed and unobserved samples and they have defined separate loss functions for each case; in our work, we allow portions of a sequence to be latent. [Teng et al. \(2020\)](#) characterized the semi-supervised VAEs via sparse latent variables; see [Mousavi et al. \(2019\)](#) for an in-depth study of additional sparse models.

Of the approaches that have been developed for handling discrete latent variables in a neural model ([Vahdat et al., 2018a,b](#); [Lorberbom et al., 2019](#), i.a.), we use the Gumbel-Softmax reparametrization ([Jang et al., 2017](#); [Maddison et al., 2016](#)). This approximates a discrete draw with logits  $\pi$  as  $\text{softmax}(\frac{\pi+g}{\tau})$ , where  $g$  is a vector of Gumbel(0, 1) draws and  $\tau$  is an annealing temperature that allows targeted behavior; its ease, customizability, and efficacy are big advantages.

### 2.2 Event Modeling

Sequential event modeling, as in this paper, can be viewed as a type of *script* or *schema* induction ([Schank and Abelson, 1977](#)) via language modeling techniques. [Mooney and DeJong \(1985\)](#) provided an early analysis of explanatory schema

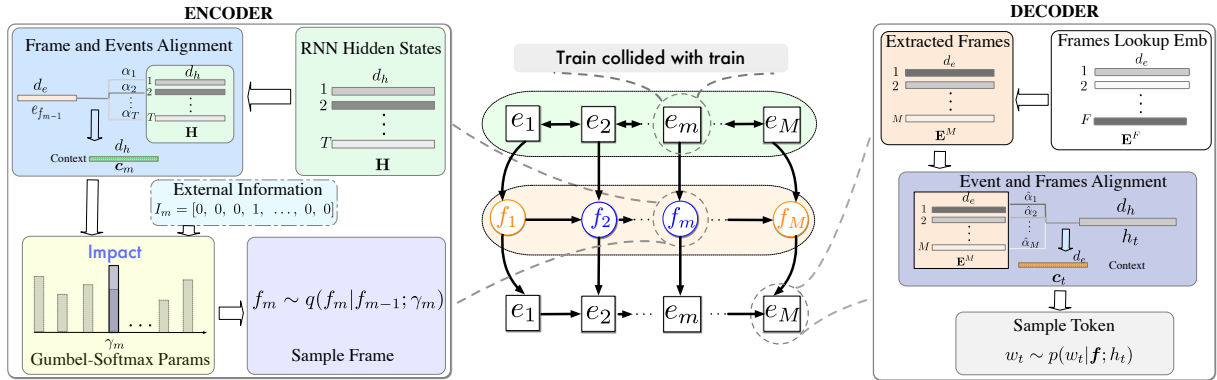


Figure 2: Our encoder (left) and decoder (right). The orange nodes mean that the frame is latent ( $I_m = \mathbf{0}$ ), while the blue nodes indicate observed frames ( $I_m$  is one-hot). In the encoder, the RNN hidden vectors are aligned with the frames to predict the next frame. The decoder utilizes the inferred frame information in the reconstruction.

generating system to process narratives, and Pichotta and Mooney (2016) applied an LSTM-based model to predict the event arguments. Modi (2016) proposed a neural network model to predict randomly missing events, while Rudinger et al. (2015) showed how neural language modeling can be used for sequential event prediction. Weber et al. (2018a) and Ding et al. (2019) used tensor-based decomposition methods for event representation. Weber et al. (2020) studied causality in event modeling via a latent neural language model.

Previous work has also examined how to incorporate or learn various forms of semantic representations while modeling events. Cheung et al. (2013) introduced an HMM-based model to explain event sequences via latent frames. Materna (2012), Chambers (2013) and Bamman et al. (2013) provided structured graphical models to learn event models over syntactic dependencies; Ferraro and Van Durme (2016) unified and generalized these approaches to capture varying levels of semantic forms and representation. Kallmeyer et al. (2018) proposed a Bayesian network based on a hierarchical dependency between syntactic dependencies and frames. Ribeiro et al. (2019) provided an analysis of clustering predicates and their arguments to infer semantic frames.

Variational autoencoders and attention networks (Kingma and Welling, 2013; Bahdanau et al., 2014), allowed Bisk et al. (2019) to use RNNs with attention to capture the abstract and concrete script representations. Weber et al. (2018b) came up with a recurrent autoencoder model (HAQAE), which used vector-quantization to learn hierarchical dependencies among discrete latent variables and an observed event sequence. Kiyomaru et al. (2019) suggested generating next events using a condi-

tional VAE-based model.

In another thread of research, Chen et al. (2018) utilized labels in conjunction with latent variables, but unlike Weber et al. (2018b), their model’s latent variables are conditionally independent and do not form a hierarchy. Sønderby et al. (2016) proposed a sequential latent structure with Gaussian latent variables. Liévin et al. (2019), similar to our model structure, provided an analysis of hierarchical relaxed categorical sampling but for the unsupervised settings.

### 3 Method

Our focus in this paper is modeling sequential event structure. In this section, we describe our variational autoencoder model, and demonstrate how partially-observed external knowledge can be injected into the learning process. We provide an overview of our joint model in §3.1 and Fig. 2. Our model operates on sequences of events: it consists of an encoder (§3.2) that encodes the sequence of events as a new sequence of frames (higher-level, more abstract representations), and a decoder (§3.3) that learns how to reconstruct the original sequence of events from the representation provided by the encoder. During training (§3.4), the model can make use of partially-observed sequential knowledge to enrich the representations produced by the encoder. In §3.5 we summarize the novel aspects of our model.

#### 3.1 Model Setup

We define each document as a sequence of  $M$  events. In keeping with previous work on event representation, each event is represented as a lexicalized 4-tuple: the core event predicate (verb), two main arguments (subject and object), and event

modifier (if applicable) (Pichotta and Mooney, 2016; Weber et al., 2018b). For simplicity, we can write each document as a sequence of  $T$  words  $\mathbf{w} = \{w_t\}_{t=1}^T$ , where  $T = 4M$  and each  $w_t$  is from a vocabulary of size  $V$ .<sup>1</sup> Fig. 1 gives an example of 3 events: during learning (but not testing) our model would have access to some, but not all, frames to lightly guide training (in this case, the first two frames but not the third).

While lexically rich, this 4-tuple representation is limited in the knowledge that can be directly encoded. Therefore, our model assumes that each document  $\mathbf{w}$  can be explained jointly with a collection of  $M$  random variables  $f_m$ :  $\mathbf{f} = \{f_m\}_{m=1}^M$ . The joint probability for our model factorizes as

$$p(\mathbf{w}, \mathbf{f}) = \prod_{t=1}^T p(w_t | \mathbf{f}, w_{<t}) \prod_{m=1}^M p(f_m | f_{m-1}). \quad (3)$$

For event modeling, each  $f_m$  represents a semantic frame. We assume there are  $F$  unique frames and let  $f_m$  be a discrete variable indicating which frame, if any, was triggered by event  $m$ .<sup>2</sup>

In the general case,  $\mathbf{f}$  is completely unobserved, and so inference for this model requires marginalizing over  $\mathbf{f}$ : when  $F \gg 1$ , optimizing the likelihood is intractable. We follow amortized variational inference (Kingma and Welling, 2013) as an alternative approach and use an ancestral sampling technique to compute it. We define  $q(\mathbf{f} | \mathbf{w})$  as the variational distribution over  $\mathbf{f}$ , which can be thought of as stochastically encoding  $\mathbf{w}$  as  $\mathbf{f}$ .

Our method is semi-supervised, so we follow Kingma et al. (2014), Chen et al. (2018) and Ye et al. (2020) and optimize a weighted variant of the evidence lower bound (ELBO),

$$\mathcal{L} = \underbrace{\mathbb{E}_{q(\mathbf{f} | \mathbf{w})} \log p(\mathbf{w} | \mathbf{f})}_{\text{Reconstruction term}} + \underbrace{\alpha_q \mathbb{E}_{q(\mathbf{f} | \mathbf{w})} \log \frac{p(\mathbf{f})}{q(\mathbf{f} | \mathbf{w})}}_{\text{KL term}},$$

$$+ \underbrace{\alpha_c \mathcal{L}_c(q(\mathbf{f} | \mathbf{w}))}_{\text{Supervised classification term}}, \quad (4)$$

where  $\mathcal{L}_c(q(\mathbf{f} | \mathbf{w}))$  is a classification objective that encourages  $q$  to predict the frames that actually were observed, and  $\alpha_q$  and  $\alpha_c$  are empirically-set to give different weight to the KL vs. classification terms. We define  $\mathcal{L}_c$  in §3.4. The reconstruc-

tion term learns to generate the observed events  $\mathbf{w}$  across all valid encodings  $\mathbf{f}$ , while the KL term uses the prior  $p(\mathbf{f})$  to regularize  $q$ .

Optimizing Eq. (4) is in general intractable, so we sample  $S$  chains of variables  $\mathbf{f}^{(1)}, \dots, \mathbf{f}^{(S)}$  from  $q(\mathbf{f} | \mathbf{w})$  and approximate Eq. (4) as

$$\mathcal{L} \approx \frac{1}{S} \sum_s \left[ \log p(\mathbf{w} | \mathbf{f}^{(s)}) + \alpha_q \log \frac{p(\mathbf{f}^{(s)})}{q(\mathbf{f}^{(s)} | \mathbf{w})} + \alpha_c \mathcal{L}_c(q(\mathbf{f}^{(s)} | \mathbf{w})) \right]. \quad (5)$$

As our model is designed to allow the injection of external knowledge  $\mathbf{I}$ , we define the variational distribution as  $q(\mathbf{f} | \mathbf{w}; \mathbf{I})$ . In our experiments,  $I_m$  is a binary vector encoding which (if any) frame is observed for event  $m$ .<sup>3</sup> For example in Fig. 1, we have  $I_1 = 1$  and  $I_3 = 0$ . We define

$$q(\mathbf{f} | \mathbf{w}; \mathbf{I}) = \prod_{m=1}^M q(f_m | f_{m-1}, I_m, \mathbf{w}). \quad (6)$$

We broadly refer to Eq. (6) as our **encoder**; we detail this in §3.2. In §3.3 we describe how we compute the reconstruction term, and in §3.4 we provide our semi-supervised training procedure.

### 3.2 Encoder

The reconstruction term relies on the frame samples given by the encoder. As discussed above though, we must be able to draw chains of variables  $\mathbf{f}$ , by iteratively sampling  $f_m \sim q(\cdot | f_{m-1}, \mathbf{w}; \mathbf{I})$ , in a way that allows the external knowledge  $\mathbf{I}$  to *guide*, but not *restrict*,  $\mathbf{f}$ . This is a deceptively difficult task, as the encoder must be able to take the external knowledge into account in a way that neither prevents nor harms back-propagation and learning. We solve this problem by learning to compute a good representation  $\gamma_m$  for each event, and sampling the current frame  $f_m$  from a Gumbel-Softmax distribution (Jang et al., 2017) parametrized by  $\gamma_m$ .

Alg. 1 gives a detailed description of our encoder. We first run our event sequence through a recurrent network (like an RNN or bi-LSTM); if  $\mathbf{w}$  is  $T$  tokens long, this produces  $T$  hidden representations, each of size  $d_h$ . Let this collection be  $\mathbf{H} \in \mathbb{R}^{T \times d_h}$ . Our encoder proceeds iteratively over each of the  $M$  events as follows: given the previous sampled frame  $f_{m-1}$ , the encoder first computes a weighted embedding  $e_{f_{m-1}}$  of this previous frame (line 1).

<sup>3</sup>If a frame is observed, then  $I_m$  is a one-hot vector where the index of the observed frame is 1. Otherwise  $I_m = \vec{0}$ .

<sup>1</sup>Table 4 in the Appendix provides all the notations.

<sup>2</sup>Our model is theoretically adaptable to making use of multiple frames, though we assume each event triggers at most one frame.

---

**Algorithm 1** Encoder: The following algorithm shows how we compute the next frame  $f_m$  given the previous frame  $f_{m-1}$ . We compute and return a hidden frame representation  $\gamma_m$ , and  $f_m$  via a continuous Gumbel-Softmax reparametrization.

---

**Input:** previous frame  $f_{m-1}$ ,  $\triangleright f_{m-1} \in \mathbb{R}^F$   
current frame observation ( $I_m$ ),

encoder GRU hidden states  $\mathbf{H} \in \mathbb{R}^{T \times d_h}$ .

**Parameters:**  $W_{\text{in}} \in \mathbb{R}^{d_h \times d_e}$ ,  $W_{\text{out}} \in \mathbb{R}^{F \times d_h}$ ,  
frames embeddings  $\mathbf{E}^F \in \mathbb{R}^{F \times d_e}$

**Output:**  $f_m, \gamma_m$

- 1:  $e_{f_{m-1}} = f_{m-1}^\top \mathbf{E}^F$
  - 2:  $\alpha \leftarrow \text{Softmax}(\mathbf{H}W_{\text{in}}e_{f_{m-1}}^\top) \triangleright$  Attn. Scores
  - 3:  $\mathbf{c}_m \leftarrow \mathbf{H}^\top \alpha \triangleright$  Context Vector
  - 4:  $\gamma'_m \leftarrow W_{\text{out}}(\tanh(W_{\text{in}}e_{f_{m-1}}) + \tanh(\mathbf{c}_m))$
  - 5:  $\gamma_m \leftarrow \gamma_m + \|\gamma_m\| I_m \triangleright$  Observation
  - 6:  $q(f_m|f_{m-1}) \leftarrow \text{GumbelSoftmax}(\gamma_m)$
  - 7:  $f_m \sim q(f_m|f_{m-1}) \triangleright f_m \in \mathbb{R}^F$
- 

Next, it calculates the similarity between  $e_{f_{m-1}}$  and RNN hidden representations and all recurrent hidden states  $\mathbf{H}$  (line 2). After deriving the attention scores, the weighted average of hidden states ( $\mathbf{c}_m$ ) summarizes the role of tokens in influencing the frame  $f_m$  for the  $m^{\text{th}}$  event (line 3). We then combine the previous frame embedding  $e_{f_{m-1}}$  and the current context vector  $\mathbf{c}_m$  to obtain a representation  $\gamma'_m$  for the  $m^{\text{th}}$  event (line 4).

While  $\gamma'_m$  may be an appropriate representation if no external knowledge is provided, our encoder needs to be able to inject any provided external knowledge  $I_m$ . Our model defines a chain of variables—some of which may be observed and some of which may be latent—so care must be taken to preserve the gradient flow within the network. We note that an initial strategy of solely using  $I_m$  instead of  $f_m$  (whenever  $I_m$  is provided) is not sufficient to ensure gradient flow. Instead, we incorporate the observed information given by  $I_m$  by adding this information to the output of the encoder logits before drawing  $f_m$  samples (line 5). This remedy motivates the encoder to *softly* increase the importance of the observed frames during the training. Finally, we draw  $f_m$  from the Gumbel-Softmax distribution (line 7).

For example, in Fig. 1, when the model knows that [IMPACT] is triggered, it increases the value of [IMPACT] in  $\gamma_m$  to encourage [IMPACT] to be sampled, but it does not prevent other frames from being sampled. On the other hand, when a frame is

not observed in training, such as for the third event ([KILLING]),  $\gamma_m$  is *not* adjusted.

Since each draw  $f_m$  from a Gumbel-Softmax is a simplex vector, given learnable frame embeddings  $\mathbf{E}^F$ , we can obtain an aggregate frame representation  $e_m$  by calculating  $e_m = f_m^\top \mathbf{E}^F$ . This can be thought of as roughly extracting row  $m$  from  $\mathbf{E}^F$  for low entropy draws  $f_m$ , and using many frames in the representation for high entropy draws. Via the temperature hyperparameter, the Gumbel-Softmax allows us to control the entropy.

### 3.3 Decoder

Our decoder (Alg. 2) must be able to reconstruct the input event token sequence from the frame representations  $\mathbf{f} = (f_1, \dots, f_M)$  computed by the encoder. In contrast to the encoder, the decoder is relatively simple: we use an auto-regressive (left-to-right) GRU to produce hidden representations  $z_t$  for each token we need to reconstruct, but we enrich that representation via an attention mechanism over  $\mathbf{f}$ . Specifically, we use both  $\mathbf{f}$  and the same learned frame embeddings  $\mathbf{E}^F$  from the encoder to compute inferred, contextualized frame embeddings as  $\mathbf{E}^M = \mathbf{f}\mathbf{E}^F \in \mathbb{R}^{M \times d_e}$ . For each output token (time step  $t$ ), we align the decoder GRU hidden state  $h_t$  with the rows of  $\mathbf{E}^M$  (line 1). After calculating the scores for each frame embedding, we obtain the output context vector  $\mathbf{c}_t$  (line 2), which is used in conjunction with the hidden state of the decoder  $z_t$  to generate the  $w_t$  token (line 5). In Fig. 1, the collection of all the three frames and the tokens from the first event will be used to predict the *Train* token from the second event.

### 3.4 Training Process

We now analyze the different terms in Eq. (5). In our experiments, we have set the number of samples  $S$  to be 1. From Eq. (6), and using the sampled sequence of frames  $f_1, f_2, \dots, f_M$  from our encoder, we approximate the reconstruction term as  $\mathcal{L}_w = \sum_t \log p(w_t|f_1, f_2, \dots, f_M; z_t)$ , where  $z_t$  is the decoder GRU’s hidden representation after having reconstructed the previous  $t - 1$  words in the event sequence.

Looking at the KL-term, we define the *prior* frame-to-frame distribution as  $p(f_m|f_{m-1}) = 1/F$ , and let the variational distribution capture the dependency between the frames. A similar type of strategy has been exploited successfully by Chen et al. (2018) to make computation simpler. We see the computational benefits

**Algorithm 2** Decoder: To (re)generate each token in the event sequence, we compute an attention  $\hat{\alpha}$  over the sequence of frame random variables  $\mathbf{f}$  (from Alg. 1). This attention weights each frame’s contribution to generating the current word.

---

**Input:**  $\mathbf{E}^M \in \mathbb{R}^{M \times d_e}$  (computed as  $\mathbf{f}\mathbf{E}^F$ )  
decoder’s current hidden state  $z_t \in \mathbb{R}^{d_h}$   
**Parameters:**  $\hat{W}_{\text{in}} \in \mathbb{R}^{d_e \times d_h}$ ,  $\hat{W}_{\text{out}} \in \mathbb{R}^{V \times d_e}$ ,  $\mathbf{E}^F$   
**Output:**  $w_t$

- 1:  $\hat{\alpha} \leftarrow \text{Softmax}(\mathbf{E}^M \hat{W}_{\text{in}} z_t)$   $\triangleright$  Attn. Scores
- 2:  $\mathbf{c}_t \leftarrow \mathbf{E}^M \hat{\alpha}$   $\triangleright$  Context Vector
- 3:  $g \leftarrow \hat{W}_{\text{out}} (\tanh(\hat{W}_{\text{in}} z_t) + \tanh(\mathbf{c}_t))$
- 4:  $p(w_t | \mathbf{f}; z_t) \propto \exp(g)$
- 5:  $w_t \sim p(w_t | \mathbf{f}; z_t)$

---

of a uniform prior: splitting the KL term into  $\mathbb{E}_{q(\mathbf{f}|w)} \log p(\mathbf{f}) - \mathbb{E}_{q(\mathbf{f}|w)} \log q(\mathbf{f}|w)$  allows us to neglect the first term. For the second term, we normalize the Gumbel-Softmax logits  $\gamma_m$ , i.e.,  $\gamma_m = \text{Softmax}(\gamma_m)$ , and compute

$$\mathcal{L}_q = -\mathbb{E}_{q(\mathbf{f}|w)} \log q(\mathbf{f}|w) \approx -\sum_m \gamma_m^\top \log \gamma_m.$$

$\mathcal{L}_q$  encourages the entropy of the variational distribution to be high which makes it hard for the encoder to distinguish the true frames from the wrong ones. We add a *fixed and constant* regularization coefficient  $\alpha_q$  to decrease the effect of this term (Bowman et al., 2015). We define the classification loss as  $\mathcal{L}_c = -\sum_{I_m > 0} I_m^\top \log \gamma_m$ , to encourage  $q$  to be good at predicting any frames that were actually observed. We weight  $\mathcal{L}_c$  by a fixed coefficient  $\alpha_c$ . Summing these losses together, we arrive at our objective function:

$$\mathcal{L} = \mathcal{L}_w + \alpha_q \mathcal{L}_q + \alpha_c \mathcal{L}_c. \quad (7)$$

### 3.5 Relation to prior event modeling

A number of efforts have leveraged frame induction for event modeling (Cheung et al., 2013; Chambers, 2013; Ferraro and Van Durme, 2016; Kallmeyer et al., 2018; Ribeiro et al., 2019). These methods are restricted to explicit connections between events and their corresponding frames; they do not capture all the possible connections between the observed events and frames. Weber et al. (2018b) proposed a hierarchical unsupervised attention structure (HAQAE) that corrects for this. HAQAE uses vector quantization (Van Den Oord et al., 2017) to capture sequential event structure via tree-based latent variables.

Our model is related to HAQAE (Weber et al., 2018b), though with important differences. While HAQAE relies on unsupervised deterministic inference, we aim to incorporate the frame information in a softer, guided fashion. The core differences are: our model supports partially-observed frame sequences (i.e., semi-supervised learning); the linear-chain connection among the event variables in our model is simpler than the tree-based structure in HAQAE; while both works use attention mechanisms in the encoder and decoder, our attention mechanism is based on addition rather than concatenation; and our handling of latent discrete variables is based on the Gumbel-Softmax reparametrization, rather than vector quantization. We discuss these differences further in Appendix C.1.

## 4 Experimental Results

We test the performance of our model on a portion of the Concretely Annotated Wikipedia dataset (Ferraro et al., 2014), which is a dump of English Wikipedia that has been annotated with the outputs of more than a dozen NLP analytics; we use this as it has readily-available FrameNet annotations provided via SemaFor (Das et al., 2014). Our training data has 457k documents, our validation set has 16k documents, and our test set has 21k documents. More than 99% of the frames are concentrated in the first 500 most common frames, so we set  $F = 500$ . Nearly 15% of the events did not have any frame, many of which were due to auxiliary/modal verb structures; as a result, we did not include them. For all the experiments, the vocabulary size ( $V$ ) is set as 40k and the number of events ( $M$ ) is 5; this is to maintain comparability with HAQAE. For the documents that had more than 5 events, we extracted the first 5 events that had frames. For both the validation and test datasets, we have set  $I_m = 0$  for all the events; frames are only observed during training.

Documents are fed to the model as a sequence of events with verb, subj, object and modifier elements. The events are separated with a special separating <TUP> token and the missing elements are represented with a special NONE token. In order to facilitate semi-supervised training and examine the impact of frame knowledge, we introduce a user-set value  $\epsilon$ : in each document, for event  $m$ , the true value of the frame is preserved in  $I_m$  with probability  $\epsilon$ , while with probability  $1 - \epsilon$  we set  $I_m = 0$ . This  $\epsilon$  is set and fixed prior to each experiment. For

Model	$\epsilon$	PPL	
		Valid	Test
RNNLM	-	61.34 $\pm$ 2.05	61.80 $\pm$ 4.81
RNNLM+ROLE	-	66.07 $\pm$ 0.40	60.99 $\pm$ 2.11
HAQAE	-	24.39 $\pm$ 0.46	21.38 $\pm$ 0.25
Ours	0.0	41.18 $\pm$ 0.69	36.28 $\pm$ 0.74
Ours	0.2	38.52 $\pm$ 0.83	33.31 $\pm$ 0.63
Ours	0.4	37.79 $\pm$ 0.52	33.12 $\pm$ 0.54
Ours	0.5	35.84 $\pm$ 0.66	31.11 $\pm$ 0.85
Ours	0.7	24.20 $\pm$ 1.07	21.19 $\pm$ 0.76
Ours	0.8	23.68 $\pm$ 0.75	20.77 $\pm$ 0.73
Ours	0.9	<b>22.52 <math>\pm</math> 0.62</b>	<b>19.84 <math>\pm</math> 0.52</b>

(a) Validation and test per-word perplexities (lower is better). We always outperform RNNLM and RNNLM+ROLE, and outperform HAQAE when automatically extracted frames are sufficiently available during training ( $\epsilon \in \{0.7, 0.8, 0.9\}$ ).

Model	$\epsilon$	Inv Narr Cloze			
		Wiki		NYT	
		Valid	Test	Valid	Test
RNNLM	-	20.33 $\pm$ 0.56	21.37 $\pm$ 0.98	18.11 $\pm$ 0.41	17.86 $\pm$ 0.80
RNNLM+ROLE	-	19.57 $\pm$ 0.68	19.69 $\pm$ 0.97	17.56 $\pm$ 0.10	17.95 $\pm$ 0.25
HAQAE	-	29.18 $\pm$ 1.40	24.88 $\pm$ 1.35	20.5 $\pm$ 1.31	22.11 $\pm$ 0.49
Ours	0.0	43.80 $\pm$ 2.93	45.75 $\pm$ 3.47	29.40 $\pm$ 1.17	28.63 $\pm$ 0.37
Ours	0.2	45.78 $\pm$ 1.53	44.38 $\pm$ 2.10	29.50 $\pm$ 1.13	29.30 $\pm$ 1.45
Ours	0.4	<b>47.65 <math>\pm</math> 3.40</b>	<b>47.88 <math>\pm</math> 3.59</b>	<b>30.01 <math>\pm</math> 1.27</b>	<b>30.61 <math>\pm</math> 0.37</b>
Ours	0.5	42.38 $\pm$ 2.41	40.18 $\pm$ 0.90	29.36 $\pm$ 1.58	29.95 $\pm$ 0.97
Ours	0.7	38.40 $\pm$ 1.20	39.08 $\pm$ 1.55	29.15 $\pm$ 0.95	30.13 $\pm$ 0.66
Ours	0.8	39.48 $\pm$ 3.02	38.96 $\pm$ 2.75	29.50 $\pm$ 0.30	30.33 $\pm$ 0.81
Ours	0.9	35.61 $\pm$ 0.62	35.56 $\pm$ 1.70	28.41 $\pm$ 0.29	29.01 $\pm$ 0.84

(b) Inverse Narrative Cloze scores (higher is better), averaged across 3 runs, with standard deviation reported. Some frame observation ( $\epsilon = 0.4$ ) is most effective across Wikipedia and NYT, though we outperform our baselines, including the SOTA, at any level of frame observation. For the NYT dataset, we first trained the model on the Wikipedia dataset and then did the tests on the NYT valid and test inverse narrative cloze datasets.

Table 1: Validation and test results for per-word perplexity (Table 1a: lower is better) and inverse narrative cloze accuracy (Table 1b: higher is better). Recall that  $\epsilon$  is the (average) percent of frames observed during *training* though during evaluation *no* frames are observed.

all the experiments we set  $\alpha_q$  and  $\alpha_c$  as 0.1, found empirically on the validation data.

**Setup** We represent words by their pretrained Glove 300 embeddings and used gradient clipping at 5.0 to prevent exploding gradients. We use a two layer of bi-directional GRU for the encoder, and a two layer uni-directional GRU for the decoder (with a hidden dimension of 512 for both). See Appendix B for additional computational details.<sup>4</sup>

**Baselines** In our experiments, we compare our proposed methods against the following methods:

- **RNNLM**: We report the performance of a sequence to sequence language model with the same structure used in our own model. A Bi-directional GRU cell with two layers, hidden dimension of 512, gradient clipping at 5 and Glove 300 embeddings to represent words.
- **RNNLM+ROLE** (Pichotta and Mooney, 2016): This model has the same structure as RNNLM, but the role for each token (verb, subject, object, modifier) as a learnable embedding vector is concatenated to the token embeddings and then it is fed to the model. The embedding dimension for roles is 300.
- **HAQAE** (Weber et al., 2018b) This work is the most similar to ours. For fairness, we seed HAQAE with the same dimension GRUs and pretrained embeddings.

<sup>4</sup><https://github.com/mmrezaee/SSDVAE>

## 4.1 Evaluations

To measure the effectiveness of our proposed model for event representation, we first report the perplexity and Inverse Narrative Cloze metrics.

**Perplexity** We summarize our per-word perplexity results in Table 1a, which compares our event chain model, with varying  $\epsilon$  values, to the three baselines.<sup>5</sup> Recall that  $\epsilon$  refers to the (average) percent of frames observed during *training*. During evaluation *no* frames are observed; this ensures a fair comparison to our baselines.

As clearly seen, our model outperforms other baselines across both the validation and test datasets. We find that increasing the observation probability  $\epsilon$  consistently yields performance improvement. For any value of  $\epsilon$  we outperform RNNLM and RNNLM+ROLE. HAQAE outperforms our model for  $\epsilon \leq 0.5$ , while we outperform HAQAE for  $\epsilon \in \{0.7, 0.8, 0.9\}$ . This suggests that while the tree-based latent structure can be helpful when external, semantic knowledge is *not* sufficiently available during training, a simpler linear structure can be successfully guided by that knowledge when it is available. Finally, recall that the external frame annotations are automatically provided, without human curation: this suggests that our model does not require perfectly, curated annotations. These observations support the hypothesis that frame observations, in conjunction with latent variables, provide a benefit to event modeling.

<sup>5</sup>In our case, perplexity provides an indication of the model’s ability to predict the next event and arguments.

Tokens	$\beta_{\text{enc}}$ (Frames Given Tokens)					
kills	KILLING	DEATH	HUNTING_SUCCESS_OR_FAILURE	HIT_TARGET	ATTACK	
paper	SUBMITTING_DOCUMENTS	SUMMARIZING	SIGN	DECIDING	EXPLAINING_THE_FACTS	
business	COMMERCE_PAY	COMMERCE_BUY	EXPENSIVENESS	RENTING	REPORTING	
Frames	$\beta_{\text{dec}}$ (Tokens Given Frames)					
CAUSATION	raise	rendered	caused	induced	brought	
PERSONAL_RELATIONSHIP	dated	dating	married	divorced	widowed	
FINISH_COMPETITION	lost	compete	won	competed	competes	

Table 2: Results for the outputs of the attention layer, the upper table shows the  $\beta_{\text{enc}}$  and the bottom table shows the  $\beta_{\text{dec}}$ , when  $\epsilon = 0.7$ . Each row shows the top 5 words for each clustering.

**Inverse Narrative Cloze** This task has been proposed by Weber et al. (2018b) to evaluate the ability of models to classify the legitimate sequence of events over detractor sequences. For this task, we have created two Wiki-based datasets from our validation and test datasets, each with 2k samples. Each sample has 6 options in which the first events are the same and only one of the options represents the actual sequence of events. All the options have a fixed size of 6 events and the one that has the lowest perplexity is selected as the correct one. We also consider two NYT inverse narrative cloze datasets that are publicly available.<sup>6</sup> All the models are trained on the Wiki dataset and then classifications are done on the NYT dataset (no NYT training data was publicly available).

Table 1b presents the results for this task. Our method tends to achieve a superior classification score over all the baselines, even for small  $\epsilon$ . Our model also yields performance improvements on the NYT validation and test datasets. We observe that the inverse narrative cloze scores for the NYT datasets is almost independent from the  $\epsilon$ . We suspect this due to the different domains between training (Wikipedia) and testing (newswire).

Note that while our model’s perplexity improved monotonically as  $\epsilon$  increased, we do not see monotonic changes, with respect to  $\epsilon$ , for this task. By examining computed quantities from our model, we observed both that a high  $\epsilon$  resulted in very low entropy attention and that frames very often attended to the verb of the event—it learned this association despite never being explicitly directly to. While this is a benefit to localized next word prediction (i.e., perplexity), it is detrimental to inverse narrative cloze. On the other hand, lower  $\epsilon$  resulted in slightly higher attention entropy, suggesting that less peaky attention allows the model to capture more of the entire event sequence and improve global coherence.

<sup>6</sup><https://git.io/Jkm46>

## 4.2 Qualitative Analysis of Attention

To illustrate the effectiveness of our proposed attention mechanism, in Table 2 we show the most likely frames given tokens ( $\beta_{\text{enc}}$ ), and tokens given frames ( $\beta_{\text{dec}}$ ). We define  $\beta_{\text{enc}} = W_{\text{out}} \tanh(\mathbf{H}^T)$  and  $\beta_{\text{dec}} = \hat{W}_{\text{out}} \tanh(\mathbf{E}^{M^T})$  where  $\beta_{\text{enc}} \in \mathbb{R}^{F \times T}$  provides a *contextual* token-to-frame soft clustering matrix for each document and analogously  $\beta_{\text{dec}} \in \mathbb{R}^{V \times M}$  provides a frame-to-word soft-clustering contextualized in part based on the inferred frames. We argue that these clusters are useful for analyzing and interpreting the model and its predictions. Our experiments demonstrate that the frames in the *encoder* (Table 2, top) mostly attend to the verbs and similarly the decoder utilizes expected and reasonable frames to predict the next verb. Note that we have not restricted the frames and tokens connection: the attention mechanism makes the ultimate decision for these connections.

We note that these clusters are a result of our attention mechanisms. Recall that in both the encoder and decoder algorithms, after computing the context vectors, we use the *addition* of two  $\tanh(\cdot)$  functions with the goal of separating the GRU hidden states and frame embeddings (line 3). This is a different computation from the bi-linear attention mechanism (Luong et al., 2015) that applies the  $\tanh(\cdot)$  function over concatenation. Our additive approach was inspired by the neural topic modeling method from Dieng et al. (2016), which similarly uses additive factors to learn an expressive and predictive neural component *and* the classic “topics” (distributions/clusters over words) that traditional topic models excel at finding. While theoretical guarantees are beyond our scope, qualitative analyses suggests that our additive attention lets the model learn reasonable soft clusters of tokens into frame-based “topics.” See Table 6 in the Appendix for an empirical comparison and validation of our use of addition rather than concatenation in the attention mechanisms.



Model	$\epsilon$	Valid			Test		
		Acc	Prec	f1	Acc	Prec	f1
RNNLM	-	0.89	0.73	0.66	0.88	0.71	0.65
RNNLM + ROLE	-	<b>0.89</b>	0.75	0.69	<b>0.88</b>	0.74	0.68
Ours	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Ours	0.20	0.59	0.27	0.28	0.58	0.27	0.28
Ours	0.40	0.77	0.49	0.50	0.77	0.49	0.50
Ours	0.50	0.79	0.51	0.48	0.79	0.50	0.48
Ours	0.70	0.85	0.69	0.65	0.84	0.69	0.65
Ours	0.80	0.86	0.77	0.74	0.85	0.76	0.74
Ours	0.90	0.87	<b>0.81</b>	<b>0.78</b>	0.86	<b>0.81</b>	<b>0.79</b>

Table 3: Accuracy and macro precision and F1-score, averaged across three different runs. We present standard deviations in Table 5.

### 4.3 How Discriminative Is A Latent Node?

Though we develop a generative model, we want to make sure the latent nodes are capable of leveraging the frame information in the decoder. We examine this assessing the ability of one single latent node to classify the frame for an event. We repurpose the Wikipedia language modeling dataset into a new training data set with 1,643,656 samples, validation with 57,261 samples and test with 75903 samples. We used 500 frame labels. Each sample is a single event. We fixed the number of latent nodes to be one. We use RNNLM and RNNLM+ROLE as baselines, adding a linear classifier layer followed by the softplus function on top of the bidirectional GRU last hidden vectors and a dropout of 0.15 on the logits. We trained all the models with the aforementioned training dataset, and tuned the hyper parameters on the validation dataset.

We trained the RNNLM and RNNLM+ROLE baselines in a purely supervised way, whereas our model mixed supervised (discriminative) and unsupervised (generative) training. The baselines observed all of the frame labels in the training set; our model only observed frame values in training with probability  $\epsilon$ , which it predicted from  $\gamma_m$ . The parameters leading to the highest accuracy were chosen to evaluate the classification on the test dataset. The results for this task are summarized in Table 3. Our method is an attention based model which captures all the dependencies in each event to construct the latent representation, but the baselines are autoregressive models. Our encoder acts like a discriminative classifier to predict the frames, where they will later be used in the decoder to construct the events. We expect the model performance to be comparable to RNNLM and RNNLM+ROLE in terms of classification when  $\epsilon$  is high. Our model with larger  $\epsilon$  tends to achieve better performance in terms of macro precision and macro F1-score.

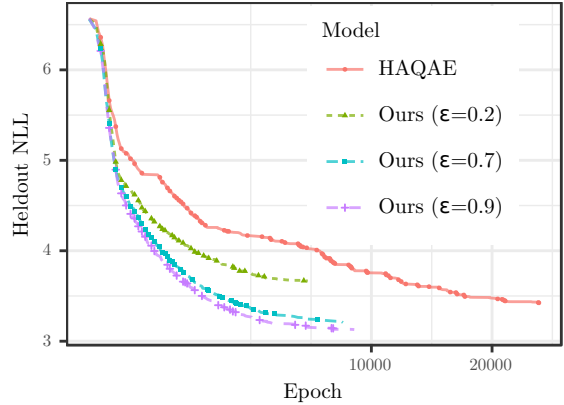


Figure 3: Validation NLL during training of our model with  $\epsilon \in \{0.2, 0.7, 0.9\}$  and HAQAE (the red curve). Epochs are displayed with a square root transform.

### 4.4 Training Speed

Our experiments show that our proposed approach converges faster than the existing HAQAE model. For fairness, we have used the same data-loader, batch size as 100, learning rate as  $10^{-3}$  and Adam optimizer. In Fig. 3, on average each iteration takes 0.2951 seconds for HAQAE and 0.2958 seconds for our model. From Fig. 3 we can see that for sufficiently high values of  $\epsilon$  our model is converging both better, in terms of negative log-likelihood (NLL), and faster—though for small  $\epsilon$ , our model still converges much faster. The reasons for this can be boiled down to utilizing Gumbel-Softmax rather than VQ-VAE, and also injection information in the form of frames jointly.

## 5 Conclusion

We showed how to learn a semi-supervised VAE with partially observed, sequential, discrete latent variables. We used Gumbel-Softmax and a modified attention to learn a highly effective event language model (low perplexity), predictor of how an initial event may progress (improved inverse narrative cloze), and a task-based classifier (outperforming fully supervised systems). We believe that future work could extend our method by incorporating other sources or types of knowledge (such as entity or “commonsense” knowledge), and by using other forms of a prior distribution, such as “plug-and-play” priors (Guo et al., 2019; Mohammedi et al., 2021; Laumont et al., 2021).

### Acknowledgements and Funding Disclosure

We would also like to thank the anonymous reviewers for their comments, questions, and sug-

gestions. Some experiments were conducted on the UMBC HPCF, supported by the National Science Foundation under Grant No. CNS-1920079. We'd also like to thank the reviewers for their comments and suggestions. This material is based in part upon work supported by the National Science Foundation under Grant Nos. IIS-1940931 and IIS-2024878. This material is also based on research that is in part supported by the Air Force Research Laboratory (AFRL), DARPA, for the KAIROS program under agreement number FA8750-19-2-1003. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either express or implied, of the Air Force Research Laboratory (AFRL), DARPA, or the U.S. Government.

## References

- Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.
- David Bamman, Brendan O'Connor, and Noah A. Smith. 2013. Learning latent personas of film characters. In *ACL*.
- Yonatan Bisk, Jan Buys, Karl Pichotta, and Yejin Choi. 2019. Benchmarking hierarchical script knowledge. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4077–4085.
- Samuel R Bowman, Luke Vilnis, Oriol Vinyals, Andrew M Dai, Rafal Jozefowicz, and Samy Bengio. 2015. Generating sentences from a continuous space. *arXiv preprint arXiv:1511.06349*.
- Nathanael Chambers. 2013. Event schema induction with a probabilistic entity-driven model. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1797–1807.
- Mingda Chen, Qingming Tang, Karen Livescu, and Kevin Gimpel. 2018. Variational sequential labelers for semi-supervised learning. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 215–226.
- Jackie Chi Kit Cheung, Hoifung Poon, and Lucy Vanderwende. 2013. Probabilistic frame induction. *NAACL HLT 2013*, pages 837–846.
- Dipanjan Das, Desai Chen, André FT Martins, Nathan Schneider, and Noah A Smith. 2014. Frame-semantic parsing. *Computational linguistics*, 40(1):9–56.
- Adji B Dieng, Chong Wang, Jianfeng Gao, and John Paisley. 2016. Topicrnn: A recurrent neural network with long-range semantic dependency. *arXiv preprint arXiv:1611.01702*.
- Xiao Ding, Kuo Liao, Ting Liu, Zhongyang Li, and Junwen Duan. 2019. Event representation learning enhanced with external commonsense knowledge. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4896–4905.
- Francis Ferraro, Adam Poliak, Ryan Cotterell, and Benjamin Van Durme. 2017. Frame-based continuous lexical semantics through exponential family tensor factorization and semantic proto-roles. In *Proceedings of the 6th Joint Conference on Lexical and Computational Semantics (\*SEM 2017)*, pages 97–103, Vancouver, Canada. Association for Computational Linguistics.
- Francis Ferraro, Max Thomas, Matthew R. Gormley, Travis Wolfe, Craig Harman, and Benjamin Van Durme. 2014. Concretely Annotated Corpora. In *4th Workshop on Automated Knowledge Base Construction (AKBC)*, Montreal, Canada.
- Francis Ferraro and Benjamin Van Durme. 2016. A unified bayesian model of scripts, frames and language. In *Thirtieth AAAI Conference on Artificial Intelligence*.
- Bichuan Guo, Yuxing Han, and Jiangtao Wen. 2019. Agem: Solving linear inverse problems via deep priors and sampling. *Advances in Neural Information Processing Systems*, 32:547–558.
- Eric Jang, Shixiang Gu, and Ben Poole. 2017. Categorical reparameterization with gumbel-softmax. In *ICLR*.
- Laura Kallmeyer, Behrang QasemiZadeh, and Jackie Chi Kit Cheung. 2018. Coarse lexical frame acquisition at the syntax–semantics interface using a latent-variable pcf model. In *Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics*, pages 130–141.
- Diederik P Kingma and Max Welling. 2013. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*.
- Durk P Kingma, Shakir Mohamed, Danilo Jimenez Rezende, and Max Welling. 2014. Semi-supervised learning with deep generative models. In *Advances in neural information processing systems*, pages 3581–3589.

- Hirokazu Kiyomaru, Kazumasa Omura, Yugo Murawaki, Daisuke Kawahara, and Sadao Kurohashi. 2019. Diversity-aware event prediction based on a conditional variational autoencoder with reconstruction. In *Proceedings of the First Workshop on Commonsense Inference in Natural Language Processing*, pages 113–122.
- Rémi Laumont, Valentin De Bortoli, Andrés Almansa, Julie Delon, Alain Durmus, and Marcelo Pereyra. 2021. Bayesian imaging using plug & play priors: when langevin meets tweedie. *arXiv preprint arXiv:2103.04715*.
- Valentin Liévin, Andrea Dittadi, Lars Maaløe, and Ole Winther. 2019. Towards hierarchical discrete variational autoencoders. In *2nd Symposium on Advances in Approximate Bayesian Inference (AABI)*.
- Guy Lorberbom, Andreea Gane, Tommi Jaakkola, and Tamir Hazan. 2019. Direct optimization through argmax for discrete variational auto-encoder. In *Advances in Neural Information Processing Systems*, pages 6203–6214.
- Minh-Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421.
- Chris J Maddison, Andriy Mnih, and Yee Whye Teh. 2016. The concrete distribution: A continuous relaxation of discrete random variables. *arXiv preprint arXiv:1611.00712*.
- Jiří Materna. 2012. LDA-Frames: an unsupervised approach to generating semantic frames. In *Computational Linguistics and Intelligent Text Processing*, pages 376–387. Springer.
- Marvin Minsky. 1974. A framework for representing knowledge. MIT-AI Laboratory Memo 306.
- Ashutosh Modi. 2016. Event embeddings for semantic script modeling. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 75–83.
- Narges Mohammadi, Marvin M Doyle, and Mujdat Cetin. 2021. Ultrasound elasticity imaging using physics-based models and learning-based plug-and-play priors. *arXiv preprint arXiv:2103.14096*.
- Raymond J Mooney and Gerald DeJong. 1985. Learning schemata for natural language processing. In *IJ-CAI*, pages 681–687.
- Seyedahmad Mousavi, Mohammad Mehdi Rezaee Taghiabadi, and Ramin Ayanzadeh. 2019. A survey on compressive sensing: Classical results and recent advancements. *arXiv preprint arXiv:1908.01014*.
- Ankur Padia, Francis Ferraro, and Tim Finin. 2018. Team UMBC-FEVER : Claim verification using semantic lexical resources. In *Proceedings of the First Workshop on Fact Extraction and VERification (FEVER)*, pages 161–165, Brussels, Belgium. Workshop at EMNLP.
- Haoruo Peng and Dan Roth. 2016. **Two discourse driven language models for semantics**. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 290–300, Berlin, Germany. Association for Computational Linguistics.
- Karl Pichotta and Raymond J Mooney. 2016. Learning statistical scripts with lstm recurrent neural networks. In *Thirtieth AAAI Conference on Artificial Intelligence*.
- Eugénio Ribeiro, Vânia Mendonça, Ricardo Ribeiro, David Martins de Matos, Alberto Sardinha, Ana Lúcia Santos, and Luísa Coheur. 2019. L2f/inesc-id at semeval-2019 task 2: Unsupervised lexical semantic frame induction using contextualized word representations. In *SemEval@NAACL-HLT*.
- Rachel Rudinger, Pushpendre Rastogi, Francis Ferraro, and Benjamin Van Durme. 2015. Script induction as language modeling. In *EMNLP*.
- Roger C Schank and Robert P Abelson. 1977. Scripts, plans, goals, and understanding: an inquiry into human knowledge structures.
- Casper Kaae Sønderby, Tapani Raiko, Lars Maaløe, Søren Kaae Sønderby, and Ole Winther. 2016. Ladder variational autoencoders. In *Advances in neural information processing systems*, pages 3738–3746.
- Michael Teng, Tuan Anh Le, Adam Scibior, and Frank Wood. 2020. Semi-supervised sequential generative models. In *Conference on Uncertainty in Artificial Intelligence*, pages 649–658. PMLR.
- Arash Vahdat, Evgeny Andriyash, and William Macready. 2018a. Dvae#: Discrete variational autoencoders with relaxed boltzmann priors. In *Advances in Neural Information Processing Systems*, pages 1864–1874.
- Arash Vahdat, William G Macready, Zhengbing Bian, Amir Khoshaman, and Evgeny Andriyash. 2018b. Dvae++: Discrete variational autoencoders with overlapping transformations. *arXiv preprint arXiv:1802.04920*.
- Aaron Van Den Oord, Oriol Vinyals, et al. 2017. Neural discrete representation learning. In *Advances in Neural Information Processing Systems*, pages 6306–6315.
- Noah Weber, Niranjana Balasubramanian, and Nathanael Chambers. 2018a. Event representations with tensor-based compositions. In *Thirty-Second AAAI Conference on Artificial Intelligence*.
- Noah Weber, Rachel Rudinger, and Benjamin Van Durme. 2020. Causal inference of script knowledge. In *Proceedings of the 2020 Conference on*

*Empirical Methods in Natural Language Processing (EMNLP).*

Noah Weber, Leena Shekhar, Niranjan Balasubramanian, and Nathanael Chambers. 2018b. Hierarchical quantized representations for script generation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3783–3792.

Rong Ye, Wenxian Shi, Hao Zhou, Zhongyu Wei, and Lei Li. 2020. Variational template machine for data-to-text generation. *arXiv preprint arXiv:2002.01127*.

Zhuosheng Zhang, Yuwei Wu, Hai Zhao, Zuchao Li, Shuailiang Zhang, Xi Zhou, and Xiang Zhou. 2020. Semantics-aware BERT for language understanding. In *AAAI*.

## A Table of Notation

Symbol	Description	Dimension
$F$	Frames vocabulary size	$\mathbb{N}$
$V$	Tokens vocabulary size	$\mathbb{N}$
$M$	Number of events, per sequence	$\mathbb{N}$
$T$	Number of words, per sequence ( $4M$ )	$\mathbb{N}$
$d_e$	Frame emb. dim.	$\mathbb{N}$
$d_h$	RNN hidden state dim.	$\mathbb{N}$
$E^F$	Learned frame emb.	$\mathbb{R}^{F \times d_e}$
$E^M$	Embeddings of sampled frames ( $E^M = fE^F$ )	$\mathbb{R}^{M \times d_e}$
$I_m$	Observed frame (external one-hot)	$\mathbb{R}^F$
$f_m$	Sampled frame (simplex)	$\mathbb{R}^F$
$\mathbf{H}$	RNN hidden states from the encoder	$\mathbb{R}^{T \times d_h}$
$z_t$	RNN hidden state from the decoder	$\mathbb{R}^{d_h}$
$W_{in}$	Learned frames-to-hidden-states weights (Encoder)	$\mathbb{R}^{d_h \times d_e}$
$W_{in}$	Learned hidden-states-to-frames weights (Decoder)	$\mathbb{R}^{d_e \times d_h}$
$W_{out}$	Contextualized frame emb. (Encoder)	$\mathbb{R}^{F \times d_h}$
$W_{out}$	Learned frames-to-words weights (Decoder)	$\mathbb{R}^{V \times d_e}$
$\alpha$	Attention scores over hidden states (Encoder)	$\mathbb{R}^T$
$\hat{\alpha}$	Attention scores over frame emb. (Decoder)	$\mathbb{R}^M$
$c_m$	Context vector (Encoder)	$\mathbb{R}^{d_h}$
$c_t$	Context vector (Decoder)	$\mathbb{R}^{d_e}$
$\gamma_m$	Gumbel-Softmax params. (Encoder)	$\mathbb{R}^F$
$w_t$	Tokens (onehot)	$\mathbb{R}^V$

Table 4: Notations used in this paper

## B Computing Infrastructure

We used the Adam optimizer with initial learning rate  $10^{-3}$  and early stopping (lack of validation performance improvement for 10 iterations). We represent events by their pretrained Glove 300 embeddings and utilized gradient clipping at 5.0 to prevent exploding gradients. The Gumbel-Softmax temperature is fixed to  $\tau = 0.5$ . We have not used dropout or batch norm on any layer. We have used two layers of Bi-directional GRU cells with a hidden dimension of 512 for the encoder module and Unidirectional GRU with the same configuration for the decoder. Each model was trained using a single GPU (a TITAN RTX RTX 2080 TI, or a Quadro 8000), though we note that neither our models nor the baselines required the full memory of any GPU (e.g., our models used roughly 6GB of GPU memory for a batch of 100 documents).

## C Additional Insights into Novelty

We have previously mentioned how our work is most similar to HAQAE (Weber et al., 2018b). In this section, we provide a brief overview of HAQAE (Appendix C.1) and then highlight differences (Fig. 4), with examples (Appendix C.2).

### C.1 Overview of HAQAE

HAQAE provides an unsupervised tree structure based on the vector quantization variational autoencoder (VQVAE) over  $M$  latent variables. Each

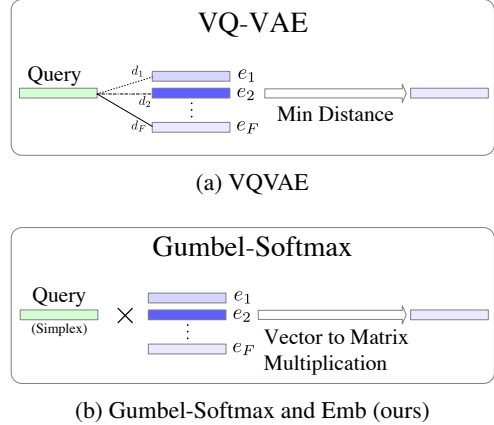


Figure 4: (a) VQ-VAE in HAQAE works based on the minimum distance between the query vector and the embeddings. (b) our approach first draws a Gumbel-Softmax sample and then extracts the relevant row by doing vector to matrix multiplication.

latent variable  $z_i$  is defined in the embedding space denoted as  $e$ . The variational distribution to approximate  $\mathbf{z} = \{z_1, z_2, \dots, z_M\}_{i=1}^M$  given tokens  $x$  is defined as follows:

$$q(\mathbf{z}|x) = q_0(z_0|x) \prod_{i=1}^{M-1} q_i(z_i|\text{parent\_of}(z_i), x).$$

The encoder calculates the attention over the input RNN hidden states  $\mathbf{h}_x$  and the parent of  $z_i$  to define  $q_i(z_i = k|z_{i-1}, x)$ :

$$\begin{cases} 1 & k = \text{argmin}_j \|g_i(x, \text{parent\_of}(z_i)) - e_{ij}\|_2 \\ 0 & \text{elsewise,} \end{cases}$$

where  $g_i(x, \text{parent\_of}(z_i))$  computes bilinear attention between  $\mathbf{h}_x$  and  $\text{parent\_of}(z_i)$ .

In this setting, the variational distribution is deterministic; right after deriving the latent query vector it will be compared with a lookup embedding table and the row with minimum distance is selected, see Fig. 4a. The decoder reconstructs the tokens as  $p(x_i|\mathbf{z})$  that calculates the attention over latent variables and the RNN hidden states. A reconstruction loss  $L_j^R$  and a ‘‘commit’’ loss (Weber et al., 2018b) loss  $L_j^C$  force  $g_j(x, \text{parent\_of}(z_i))$  to be close to the embedding referred to by  $q_i(z_i)$ . Both  $L_j^R$  and  $L_j^C$  terms rely on a deterministic mapping that is based on a nearest neighbor computation that makes it difficult to inject guiding information to the latent variable.

### C.2 Frame Vector Norm

Like HAQAE, we use embeddings  $E^F$  instead of directly using the Gumbel-Softmax frame sam-

Model	$\epsilon$	Valid			Test		
		Acc	Prec	f1	Acc	Prec	f1
RNNLM	-	0.89 $\pm$ 0.001	0.73 $\pm$ 0.004	0.66 $\pm$ 0.008	0.88 $\pm$ 0.005	0.71 $\pm$ 0.014	0.65 $\pm$ 0.006
RNNLM + ROLE	-	0.89 $\pm$ 0.004	0.75 $\pm$ 0.022	0.69 $\pm$ 0.026	0.88 $\pm$ 0.005	0.74 $\pm$ 0.017	0.68 $\pm$ 0.020
Ours	0.00	0.00 $\pm$ 0.001	0.00 $\pm$ 0.001	0.00 $\pm$ 0.000	0.00 $\pm$ 0.001	0.00 $\pm$ 0.000	0.00 $\pm$ 0.000
Ours	0.20	0.59 $\pm$ 0.020	0.27 $\pm$ 0.005	0.28 $\pm$ 0.090	0.58 $\pm$ 0.010	0.27 $\pm$ 0.050	0.28 $\pm$ 0.010
Ours	0.40	0.77 $\pm$ 0.012	0.49 $\pm$ 0.130	0.50 $\pm$ 0.010	0.77 $\pm$ 0.010	0.49 $\pm$ 0.060	0.50 $\pm$ 0.010
Ours	0.50	0.79 $\pm$ 0.010	0.51 $\pm$ 0.080	0.48 $\pm$ 0.080	0.79 $\pm$ 0.009	0.50 $\pm$ 0.080	0.48 $\pm$ 0.050
Ours	0.70	0.85 $\pm$ 0.007	0.69 $\pm$ 0.050	0.65 $\pm$ 0.040	0.84 $\pm$ 0.013	0.69 $\pm$ 0.050	0.65 $\pm$ 0.020
Ours	0.80	0.86 $\pm$ 0.003	0.77 $\pm$ 0.013	0.74 $\pm$ 0.006	0.85 $\pm$ 0.005	0.76 $\pm$ 0.008	0.74 $\pm$ 0.010
Ours	0.90	0.87 $\pm$ 0.002	0.81 $\pm$ 0.007	0.78 $\pm$ 0.006	0.86 $\pm$ 0.011	0.81 $\pm$ 0.020	0.79 $\pm$ 0.010

Table 5: Accuracy and macro precision and F1-score, averaged across 3 runs, with standard deviation reported.

$\epsilon$	Addition		Concatenation	
	Valid	Test	Valid	Test
0.0	41.18 $\pm$ 0.69	36.28 $\pm$ 0.74	48.86 $\pm$ 0.13	42.76 $\pm$ 0.08
0.2	38.52 $\pm$ 0.83	33.31 $\pm$ 0.63	48.73 $\pm$ 0.18	42.76 $\pm$ 0.06
0.4	37.79 $\pm$ 0.52	33.12 $\pm$ 0.54	49.34 $\pm$ 0.32	43.18 $\pm$ 0.21
0.5	35.84 $\pm$ 0.66	31.11 $\pm$ 0.85	49.83 $\pm$ 0.24	43.90 $\pm$ 0.32
0.7	24.20 $\pm$ 1.07	21.19 $\pm$ 0.76	52.41 $\pm$ 0.38	45.97 $\pm$ 0.48
0.8	23.68 $\pm$ 0.75	20.77 $\pm$ 0.73	55.13 $\pm$ 0.31	48.27 $\pm$ 0.21
0.9	22.52 $\pm$ 0.62	19.84 $\pm$ 0.52	58.63 $\pm$ 0.25	51.39 $\pm$ 0.34

Table 6: Validation and test per-word perplexities with bilinear-attention (lower is better).

ples. In our model definition, each frame  $f_m = [f_{m,1}, f_{m,2}, \dots, f_{m,F}]$  sampled from the Gumbel-Softmax distribution is a simplex vector:

$$0 \leq f_{m,i} \leq 1, \quad \sum_{i=1}^F f_{m,i} = 1. \quad (8)$$

So  $\|f_m\|_p = (\sum_{i=1}^F f_{m,i}^p)^{1/p} \leq F^{1/p}$ . After sampling a frame simplex vector, we multiply it to the frame embeddings matrix  $E^F$ . With an appropriate temperature for Gumbel-Softmax, the simplex would be approximately a one-hot vector and the multiplication maps the simplex vector to the embeddings space without any limitation on the norm.

## D More Results

In this section, we provide additional quantitative and qualitative results, to supplement what was presented in §4.

### D.1 Standard Deviation for Frame Prediction (§4.3)

In Table 5, we see the average results of classification metrics with their corresponding standard deviations.

### D.2 Importance of Using Addition rather than Concatenation in Attention

To demonstrate the effectiveness of our proposed attention mechanism, we compare the addition

against the concatenation (regular bilinear attention) method. We report the results on the Wikipedia dataset in Table 6. Experiments indicate that the regular bilinear attention structure with larger  $\epsilon$  obtains worse performance. These results confirm the claim that the proposed approach benefits from the addition structure.

### D.3 Generated Sentences

Recall that the reconstruction loss is

$$\mathcal{L}_w \approx \frac{1}{S} \sum_s \log p(\mathbf{w} | f_1^{(s)}, f_2^{(s)}, \dots, f_M^{(s)}; z),$$

where  $f_m^{(s)} \sim q(f_m | f_{m-1}^{(s)}, I_m, \mathbf{w})$ . Based on these formulations, we provide some examples of generated scripts. Given the seed, the model first predicts the first frame  $f_1$ , then it predicts the next verb  $v_1$  and similarly samples the tokens one-by-one. During the event generations, if the sampled token is unknown, the decoder samples again. As we see in Table 7, the generated events and frames samples are consistent which shows the ability of model in event representation.

### D.4 Inferred Frames

Using Alg. 1, we can see the frames sampled during the training and validation. In Table 8, we provide some examples of frame inferring for both training and validation examples. We observe that for training examples when  $\epsilon > 0.5$ , almost all the observed frames and inferred frames are the same. In other words, the model prediction is almost the same as the ground truth.

Interestingly, the model is more flexible in sampling the latent frames (orange ones). In Table 8a the model is estimating HAVE\_ASSOCIATED instead of the POSSESSION frame. In Table 8b, instead of PROCESS\_START we have ACTIV-

Seed	Event 1	Event 2
<b>elected taylor election at</b> [CHANGE_OF_LEADERSHIP]	served she attorney as [ASSISTANCE]	sworn she house to [COMMITMENT]
<b>graduated ford berkeley at</b> [SCRUTINY]	earned he theology in [EARNINGS_AND_LOSSES]	documented he book in [RECORDING]
<b>released album 2008 in</b> [RELEASING]	helped song lyrics none [ASSISTANCE]	made chart song with [CAUSATION]
<b>created county named and</b> [INTENTIONALLY_CREATE]	totals district population of [AMOUNTING_TO]	served district city none [ASSISTANCE]
<b>published he december on</b> [SUMMARIZING]	written book published and [TEXT_CREATION]	place book stories among [PLACING]
<b>played music show on</b> [PERFORMERS_AND_ROLES]	starred film music none [PERFORMERS_AND_ROLES]	tend lyrics music as [LIKELIHOOD]
<b>aired series canada in</b> [EXPRESSING_PUBLICLY]	features series stories none [INCLUSION]	abused characters film in [ABUSING]
<b>consists band members of</b> [INCLUSION]	belong music party to [MEMBERSHIP]	leads music genre into [CAUSATION]

Table 7: Generated scripts and the inferred frames (in brackets) from the seed event in boldface ( $\epsilon = 0.7$ )

ITY\_START and finally in Table 8d we have TAKING\_SIDES rather than SUPPORTING.

Some wrong predictions like CATASTROPHE instead of CAUSATION in Table 8c can be considered as the effect of other words like “*pressure*” in “*resulted pressure revolution in*”. In some cases like Table 8b the predicted frame BEING\_NAMED is a better choice than the ground truth APPOINTING. In the same vein FINISH\_COMPETITION is a better option rather than GETTING in Table 8f.

## D.5 Clustering

Here we provide more examples for  $\beta_{enc}$  and  $\beta_{dec}$  in Table 9. Our experiments show that by sorting the tokens in each frame, the first 20 words are mostly verbs. And among different token types, verbs are better classifiers for frames.

Event	Ground Truth	Inferred Frame
1   estimated product prices in	ESTIMATING	ESTIMATING
2   rose which billions to	CHANGE_POSITION_ON_A_SCALE	CHANGE_POSITION_ON_A_SCALE
3   had maharashtra per as_of	POSSESSION	POSSESSION
4   houses mumbai headquarters none	BUILDINGS	BUILDINGS
5   have % offices none	POSSESSION	HAVE_ASSOCIATED

(a) Example 1 (training)

Event	Ground Truth	Inferred Frame
1   competed she olympics in	WIN_PRIZE	WIN_PRIZE
2   set she championships at	CAUSE_TO_START	CAUSE_TO_START
3   named she <unk> in	APPOINTING	BEING_NAMED
4   started she had but	PROCESS_START	ACTIVITY_START
5   had she height because_of	POSSESSION	POSSESSION

(b) Example 2 (training)

Event	Ground Truth	Inferred Frame
1   arose that revolution after	COMING_TO_BE	COMING_TO_BE
2   supported who charter none	TAKING_SIDES	TAKING_SIDES
3   developed pressure classes from	PROGRESS	PROGRESS
4   remained monarchy run and	STATE_CONTINUE	STATE_CONTINUE
5   resulted pressure revolution in	CAUSATION	CATASTROPHE

(c) Example 3 (training)

Event	Ground Truth	Inferred Frame
1   features it form in	INCLUSION	INCLUSION
2   allows format provides and	PERMITTING	DENY_PERMISSION
3   provides format forum none	SUPPLY	SUPPLY
4   rely readers staff upon	RELIANCE	RELIANCE
5   support citations assertions none	SUPPORTING	TAKING_SIDES

(d) Example 4 (validation)

Event	Ground Truth	Inferred Frame
1   attended he university none	ATTENDING	ATTENDING
2   moved he 1946 in	MOTION	TRAVEL
3   married he yan in	PERSONAL_RELATIONSHIP	FORMING_RELATIONSHIPS
4   worked they moved and	BEING_EMPLOYED	BEING_EMPLOYED
5   moved they beijing to	MOTION	MOTION

(e) Example 5 (validation)

Event	Ground Truth	Inferred Frame
1   had he achievements none	POSSESSION	HAVE_ASSOCIATED
2   competed he finished and	FINISH_COMPETITION	REQUIRED_EVENT
3   finished he relay in	PROCESS_COMPLETED_STATE	ACTIVITY_DONE_STATE
4   set he three between	INTENTIONALLY_CREATE	INTENTIONALLY_CREATE
5   won he championships at	GETTING	FINISH_COMPETITION

(f) Example 6 (validation)

Table 8: Sequences of 5 events and the inferred frames during training with partial frame observation, and validation without any observation. Blue frames are observed and orange frames are latent.

		$\beta_{enc}$ (Frames Given Verbs)				
		EXECUTION	VERDICT	HIRING	RITE	BEING_OPERATIONAL
Verbs						
<b>executed</b>		ACTIVITY_START	PROCESS_START	DURATION_RELATION	ACTIVITY_DONE_STATE	RELATIVE_TIME
<b>start</b>		HOSTILE_ENCOUNTER	PARTICIPATION	SUCCESS_OR_FAILURE	QUITTING	DEATH
<b>fought</b>		BEING_EMPLOYED	WORKING_ON	HIRING	BEING_OPERATIONAL	EDUCATION_TEACHING
<b>worked</b>		CAUSATION	MANUFACTURING	INTENTIONALLY_CREATE	CAUSE_CHANGE	CREATING
<b>made</b>		DOMINATE_SITUATION	CONTROL	BOUNDARY	SEPARATING	INVADING
<b>dominated</b>		INTENTIONALLY_CREATE	CREATING	CAUSE_CHANGE	COMING_TO_BE	CAUSATION
<b>created</b>		BUILDINGS	LOCATIVE_RELATION	RESIDENCE	COMMERCE_BUY	CAUSE_EXPANSION
<b>houses</b>		PERCEPTION_EXPERIENCE	GRASP	EVIDENCE	SIMILARITY	EXPERIENCER_OBJ
<b>see</b>		ACTIVITY_START	Process_start	Secrecy_status	Sign_agreement	Activity_finish
<b>start</b>		GIVING	GETTING	BRINGING	SUPPLY	NEEDING
<b>gave</b>		TRAVERSING	UNDERGO_CHANGE	CONTROL	CHANGE_OF_LEADERSHIP	ARRIVING
<b>passed</b>		COMMERCE_SELL	COMMERCE_BUY	MANUFACTURING	PREVENT_FROM_HAVING	RELEASING
<b>sold</b>		MOTION_DIRECTIONAL	CHANGE_POSITION_ON_A_SCALE	GETTING_UP	BOUNDARY	EXPANSION
<b>rose</b>		ARRIVING	TRAVERSING	INTENTIONALLY_AFFECT	HAVE_ASSOCIATED	GETTING
<b>returned</b>		BECOMING_A_MEMBER	COLLABORATION	APPOINTING	COTHEME	COME_TOGETHER
<b>enlisted</b>		COME_TOGETHER	MEET_WITH	MAKE_ACQUAINTANCE	DISCUSSION	BOUNDARY
<b>met</b>		ACTIVITY_START	ARRIVING	PROCESS_START	UNDERGO_CHANGE	MASS_MOTION
<b>entered</b>		GETTING	COMMERCE_BUY	GIVING	CONQUERING	EARNINGS_AND_LOSSES
<b>obtained</b>		CHOOSING	VOLUBILITY	LEADERSHIP	BE_SUBSET_OF	COMING_UP_WITH
<b>chosen</b>						
Frames						
<b>BECOMING</b>		become	becomes	became	turned	goes
<b>SHOOT_PROJECTILES</b>		launched	fired	shoots	shoot	rocket
<b>JUDGMENT_COMMUNICATION</b>		criticized	charged	criticised	praised	accused
<b>SHOOT_PROJECTILES</b>		launched	fired	shoots	shoot	rocket
<b>JUDGMENT_COMMUNICATION</b>		criticized	charged	criticised	praised	accused
<b>BEING_NAMED</b>		named	entitled	known	called	honour
<b>ASSISTANCE</b>		serve	serves	served	helps	assisted
<b>CHANGE_OF_LEADERSHIP</b>		elected	elects	councillors	installed	overthrew
<b>REVEAL_SECRET</b>		exposed	revealed	disclosed	reveal	confessed
<b>TEXT_CREATION</b>		composed	authored	composes	drafted	written
<b>EXPRESSING_PUBLICLY</b>		aired	voiced	expresses	express	airs
<b>ARREST</b>		arrested	apprehended	booked	jailed	re-arrested
<b>COLLABORATION</b>		cooperate	collaborated	teamed	partnered	cooperated
<b>WIN_PRIZE</b>		competing	compete	competes	competed	mid-1950s
<b>LEADERSHIP</b>		governs	administered	administers	presides	ruled
<b>INCLUSION</b>		contains	included	includes	include	excluded
<b>SELF_MOTION</b>		sailed	walks	ran	danced	walk
<b>INGESTION</b>		eaten	drank	drink	consumed	eat
<b>TAKING_SIDES</b>		supported	sided	supporting	endorsed	endorses

Table 9: Results for the outputs of the attention layer, the upper table shows the  $\beta_{enc}$  and the bottom table shows the  $\beta_{dec}$ , when  $\epsilon = 0.7$ . Each row shows the top 5 words for each clustering.