Human-like Controllable Image Captioning with Verb-specific Semantic Roles — Supplementary Document —

The supplementary document is organized as follows:

- In Section 1, we explain the meanings of different semantic roles (*i.e.*, PropBank-style annotations) in our paper.
- In Section 2, we illustrate more visualization results generated by our CIC framework.
- In Section 3, we provide the details about each subnet component of our VSR-guided CIC model.
- In Section 4, we show the details about the merging algorithm of two different semantic structures from two VSRs.
- In Section 5, we report the details of our experimental settings.
- In Section 6, we compare the performance between the Transformer structure and Sinkhorn network in S-level SSP.

1. Meanings of Different Semantic Roles

In this paper, we mainly follow the types of semantic roles defined in the PropBank [6]. The main arguments with their semantic role meanings is listed in Table 3, including numbered arguments (*e.g.*, `Arg0`, `Arg2`)[1] and argument modifiers (*e.g.*, `COM`, `LOC`).

Although there are many kinds of arguments modifiers in the PropBank, the most common argument modifiers of the verbs in Flickr30k/COCO Entities are `LOC`, `DIR`, `GOL` and `MNR`. The meaning of them as listed as follows:

- `LOC`: Locative modifiers indicate where some action takes place.
- `DIR`: Directional modifiers show motion along some path.
- `GOL`: Goal tag is for the goal of the action of the verb.
- `MNR`: Manner modifiers specify how an action is performed.

[1]Since semantic role `Arg5` is very rare for the verbs of CIC datasets, and we omit it in Table 3.

	Role Type	Meaning
numbered args	`Arg0`	agent
	`Arg1`	patient
	`Arg2`	instrument, benefactive, attribute
	`Arg3`	starting point, benefactive, attribute
	`Arg4`	ending point
argument modifiers	`COM`	comitative
	`LOC`	locative
	`DIR`	directional
	`GOL`	goal
	`MNR`	manner
	`TMP`	temporal
	`EXT`	extent
	`REC`	reciprocals
	`PRD`	secondary predication
	`PRP`	purpose
	`PNC`	purpose not cause
	`CAU`	cause
	`DIS`	discourse
	`ADV`	adverbials
	`ADJ`	adjectival
	`MOD`	modal
	`NEG`	negation
	`LVB`	light verb

Table 3. List of the main arguments in the PropBank.

2. More Visualization Results

We illustrate more visualization results of generated image captions using the VSR corresponding to the ground truth caption in Figure 8. Meanwhile, we show more visualization results about diverse image captions conditioned on different VSRs in Figure 9. More specifically, the VSRs in the top row of images contain the same verb and different semantic role sequences; the VSRs in the bottom row of images contain a different verb or two verbs.

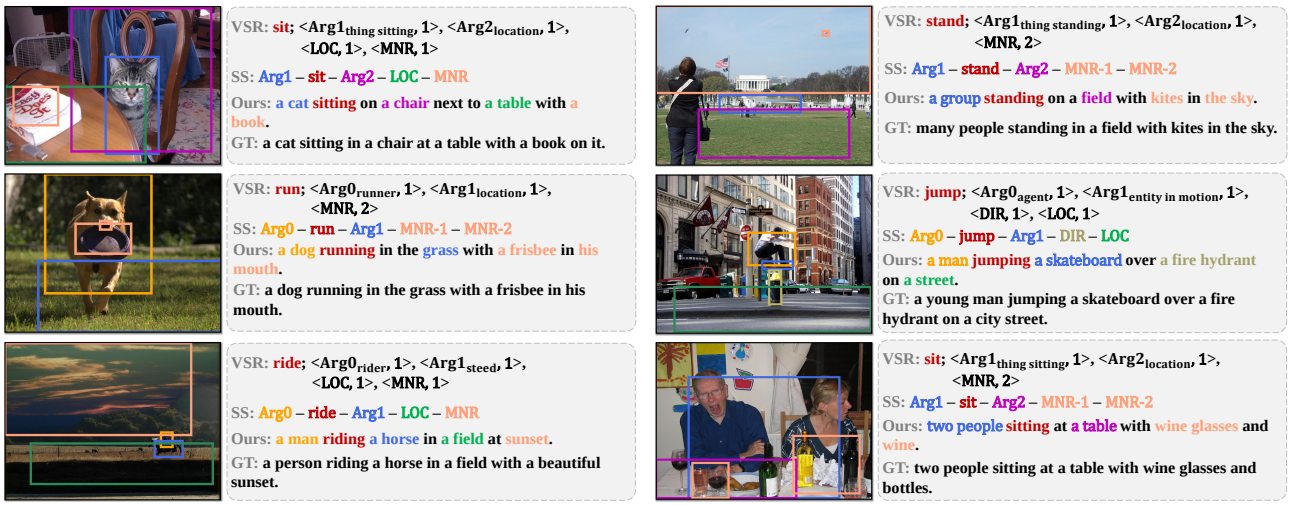

Figure 8. Additional examples of generated image captions using the VSR corresponding to the ground truth caption. **SS** denotes the learned semantic structures. Different colors show a correspondence between image regions and semantic roles. Best viewed in color.

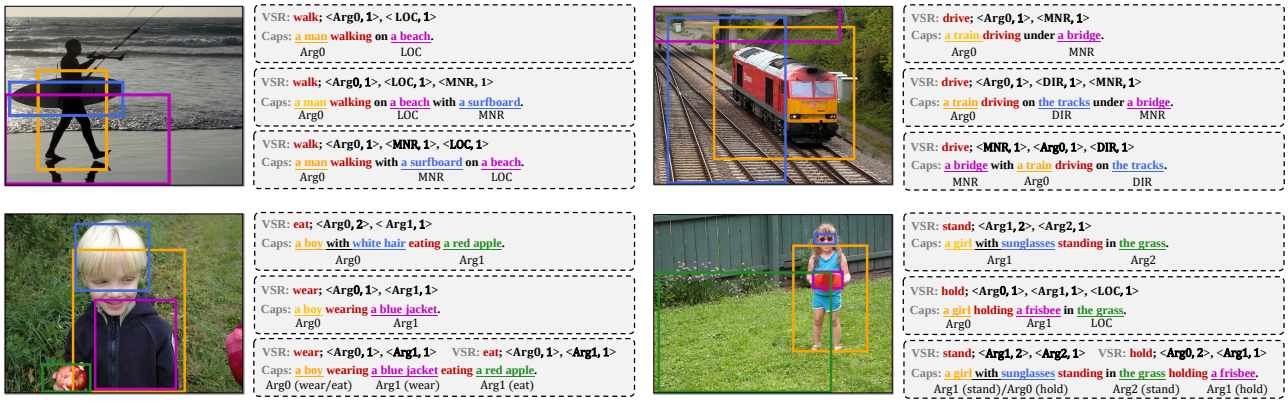

Figure 9. Additional examples of diverse image caption generation conditioned on different VSRs. The correspondences between image regions and noun phrases are indicated by different colors. Best viewed in color.

3. Details of the VSR-guided CIC Model

3.1. Grounded Semantic Role Labeling

In this grounded semantic role labeling (GSRL) step, we aim to ground each sub-role s_i in $\mathcal{VSR}$ to a proposal set $\mathcal{B}_j \in \mathcal{B}$. Specifically, we calculate the similarity score a_{ij} between sub-role s_i and proposal set $\mathcal{B}_j$ by:

$$\begin{aligned} \boldsymbol{q}_i &= [\boldsymbol{W}_v^g \Pi_v; \boldsymbol{W}_s^g \Pi_{s_i}; \bar{\boldsymbol{f}}], \\ \boldsymbol{a}_{ij} &= \mathrm{MLP}_a(\boldsymbol{W}_q^g \boldsymbol{q}_i \odot \boldsymbol{W}_f^g \bar{\boldsymbol{f}}_{\boldsymbol{j}}), \end{aligned} \tag{14}$$

where $\bar{\boldsymbol{f}} \in \mathbb{R}^{v\times 1}$ and $\bar{\boldsymbol{f}}_{\boldsymbol{j}} \in \mathbb{R}^{v\times 1}$ represent the average-pooled visual feature of proposal set $\mathcal{B}$ and $\mathcal{B}_j$. Π_v and Π_{s_i} are the one-hot embeddings for the verb v and sub-role s_i, $\boldsymbol{W}_v^g \in \mathbb{R}^{d_v\times|\mathcal{V}|}$ and $\boldsymbol{W}_s^g \in \mathbb{R}^{d_s\times|\mathcal{SR}|}$ are learnable mapping matrices, $|\mathcal{V}|$ and $|\mathcal{SR}|$ are the size of the vocabulary of verbs and semantic roles, respectively. $[;]$ is a concatenation operation. Thus, $\boldsymbol{q}_i$ is a query vector combining the verb category, semantic role type and image global features. $\boldsymbol{W}_q^g \in \mathbb{R}^{a\times(d_v+d_s+v)}$ and $\boldsymbol{W}_f^g \in \mathbb{R}^{a\times v}$ aim to transform $\boldsymbol{q}_i$ and $\bar{\boldsymbol{f}}_{\boldsymbol{j}}$ into a common space, and $\odot$ is the element-wise multiplication. Finally, a four-layer MLP maps the fused feature into a score $\boldsymbol{a}_{ij}$ between 0 and 1.

3.2. Semantic Structure Planner

S-level SSP. In the sentence-level (S-level) SSP, we utilize a three-layer Transformer encoder to encode the verb v and semantic role s_i in the input semantic role sequence $\mathcal{S}$.

$$\boldsymbol{H} = \mathrm{Transformer}_{\mathrm{enc}}\left(\{\mathrm{FC}_a(\boldsymbol{W}_v^e \Pi_v + \boldsymbol{W}_s^e \Pi_{s_i})\}\right), \tag{15}$$

where Π_v and Π_{s_i} are the one-hot embeddings for v and s_i, $\boldsymbol{W}_v^e \in \mathbb{R}^{d_e\times|\mathcal{V}|}$ and $\boldsymbol{W}_s^e \in \mathbb{R}^{d_e\times|\mathcal{SR}|}$ are learnable mapping matrices.

Then, we use a three-layer Transformer decoder to autoregressively generate semantic role sequence (including the verb). To prevent the occurrence of semantic role sequence with duplicates, we generate s_t with the highest probability $p(s_t|\mathcal{VSR})$, where s_t is in the input semantic role sequence but hasn't been generated.

$$p(s_t|\mathcal{VSR}) = \text{Transformer}_{\text{dec}}\left(\boldsymbol{H}, \boldsymbol{W}_s^e \Pi_{\mathcal{S}_{<t}}\right), \tag{16}$$

R-level SSP. Since each semantic role s_i has variable number of sub-roles (*i.e.*, n_i), we set a constant n_{max} as the maximum number of sub-roles for each semantic role. We employ the Sinkhorn operation [5] to learn a "soft" permutation matrix $\boldsymbol{P}$. For each proposal $\boldsymbol{b}_* \in \hat{\mathcal{B}}$, we encode a feature vector $\tilde{\boldsymbol{z}}_*$ by:

$$\tilde{\boldsymbol{z}}_* = \text{MLP}_b([\boldsymbol{W}_v^r \boldsymbol{f}_*; \boldsymbol{W}_c^r \Pi_{c_*}; \text{Pos}(\boldsymbol{b}_*)]), \tag{17}$$

where $\boldsymbol{f}_*$ is the detection feature (2048-d); Π_{c_*} is GloVe embedding of the region class (300-d); $\text{Pos}(\cdot)$ is a 4-d spatial encoding of $\boldsymbol{b}_*$. $\boldsymbol{W}_x^r \in \mathbb{R}^{d_v \times v}$ and $\boldsymbol{W}_c^d \in \mathbb{R}^{d_c \times |\mathcal{C}|}$ are learnable mapping matrices, $|\mathcal{C}|$ is the size of vocabulary of the detected classes, and MLP_b is a two-layer MLP to mapping the concatenated feature into $\mathbb{R}^{n_{\text{max}}}$. The position encoding function $\text{Pos}(\cdot)$ encodes the location feature: $[\frac{x_{\text{min}}}{W_I}, \frac{y_{\text{min}}}{H_I}, \frac{x_{\text{max}}}{W_I}, \frac{y_{\text{max}}}{H_I}]$, where x_{min}, y_{min}, x_{max}, y_{max} are the bounding box coordinates of proposal $\boldsymbol{b}_*$; $W_{\boldsymbol{I}}$ and $H_{\boldsymbol{I}}$ are the width and height of the image $\boldsymbol{I}$.

Then, for each proposal set $\hat{\mathcal{B}}_i \subset \hat{\mathcal{B}}$, we average-pool all the feature (*i.e.*, $\{\tilde{\boldsymbol{z}}_*\}$) of each proposal set, denoted as $\boldsymbol{z}_i$. And we concatenate all feature representations $\{\boldsymbol{z}_i\}$ to get a $n_{\text{max}} \times n_{\text{max}}$ matrix $\boldsymbol{Z}$. The square matrix $\boldsymbol{Z}$ is converted into a "soft" permutation matrix $\boldsymbol{P}$ through the Sinkhorn operator. The operator is K consecutive row-wise and column-wise normalization, as follows:

$$\begin{aligned} S^0(\boldsymbol{Z}) &= \exp(\boldsymbol{Z}), \\ S^k(\boldsymbol{Z}) &= \mathcal{N}_c(\mathcal{N}_r(S^{k-1}(\boldsymbol{Z}))), \\ \boldsymbol{P} &= S^K(\boldsymbol{Z}), \end{aligned} \tag{18}$$

where $\mathcal{N}_r(\boldsymbol{Z}) = \boldsymbol{Z} \oslash (\boldsymbol{Z}\mathbb{1}_{n_{\text{max}}}\mathbb{1}_{n_{\text{max}}}^T)$ and $\mathcal{N}_c(\boldsymbol{Z}) = \boldsymbol{Z} \oslash (\mathbb{1}_{n_{\text{max}}}\mathbb{1}_{n_{\text{max}}}^T \boldsymbol{Z})$ are the row-wise and column-wise normalization operations respectively, and $\oslash$ is the element-wise division, $\mathbb{1}_{n_{\text{max}}}$ is a column vector of n_{max} ones.

During inference, once K normalizations (we set $K = 20$ in our experiments) have been performed, the resulting "soft" permutation matrix can be converted into the final permutation matrix via the Hungarian algorithm [3].

3.3. Role-shift Captioning Model

Adaptive attention for the shifting probability. The first LSTM is firstly extended to obtain a sub-role sentinel $\boldsymbol{s}_t^g$, which models a component encoding the state of the LSTM at the end of a sub-role. The sentinel is computed as:

$$\begin{aligned} \boldsymbol{l}_t^g &= \sigma(\boldsymbol{W}_{ig}\boldsymbol{x}_t + \boldsymbol{W}_{hg}\boldsymbol{h}_{t-1}^1) \\ \boldsymbol{sr}_t^g &= \boldsymbol{l}_t^g \odot \tanh(\boldsymbol{m}_t) \end{aligned} \tag{19}$$

where $\boldsymbol{W}_{ig} \in \mathbb{R}^{d_l \times d_i}$, $\boldsymbol{W}_{hg} \in \mathbb{R}^{d_l \times d_l}$ are learnable weights, $\boldsymbol{m}_t \in \mathbb{R}^{d_l}$ is the LSTM cell memory and $\boldsymbol{x}_t \in \mathbb{R}^{d_i}$ is the input of the LSTM at time t; $\odot$ represents the Hadamard element-wise product and σ is the sigmoid function.

We then compute a compatibility score between the hidden state $\boldsymbol{h}_t^1$ and the sentinel vector $\boldsymbol{sr}_t^g$ through a single-layer neural network; analogously, we compute a compatibility score between $\boldsymbol{h}_t^1$ and the regions in $\boldsymbol{r}_t$ by:

$$\begin{aligned} \hat{\alpha}_t^g &= \boldsymbol{w}_h^T \tanh(\boldsymbol{W}_{sg}\boldsymbol{sr}_t^g + \boldsymbol{W}_g\boldsymbol{h}_t^1), \\ \hat{\boldsymbol{\alpha}}_t^r &= \boldsymbol{w}_h^T \tanh(\boldsymbol{W}_{sr}\boldsymbol{r}_t + (\boldsymbol{W}_g\boldsymbol{h}_t^1)\mathbb{1}^T), \end{aligned} \tag{20}$$

where $\mathbb{1} \in \mathbb{R}^{n_t}$ is a vector with all elements set to 1, n_t is the number of regions in $\boldsymbol{r}_t$, $\boldsymbol{w}_h^T$ is a row vector, $\boldsymbol{W}_{sg} \in \mathbb{R}^{d_a \times d_l}$, $\boldsymbol{W}_{sr} \in \mathbb{R}^{d_a \times d_v}$ and $\boldsymbol{w}_h \in \mathbb{R}^{d_a}$ are learnable mapping matrices.

And then we renormalize the attention weight for sub-role sentinel $\boldsymbol{sr}_t^g$ over attention weights for the sentinel vector $\boldsymbol{sr}_t^g$ and the regions in $\boldsymbol{r}_t$:

$$\alpha_t^g = \frac{\exp \hat{\alpha}_t^g}{\exp \hat{\alpha}_t^g + \sum_i \exp \hat{\boldsymbol{\alpha}}_{ti}^r}, \tag{21}$$

where $\hat{\boldsymbol{\alpha}}_{ti}^r$ indicates the i-th element in $\hat{\boldsymbol{\alpha}}_t^r$.

Adaptive attention for the context feature. To further distinguish the textual and visual words, we build an adaptive attention mechanism with a visual sentinel [4]. The visual sentinel vector models a component which the model can fall back on when it chooses to not attend regions in $\boldsymbol{r}_t$. Analogously to Eq. (19), it is defined as:

$$\begin{aligned} \boldsymbol{l}_t^v &= \sigma(\boldsymbol{W}_{is}\boldsymbol{x}_t + \boldsymbol{W}_{hs}\boldsymbol{h}_{t-1}^1), \\ \boldsymbol{sr}_t^v &= \boldsymbol{l}_t^v \odot \tanh(\boldsymbol{m}_t), \end{aligned} \tag{22}$$

where $\boldsymbol{W}_{is} \in \mathbb{R}^{d_l \times d_i}$ and $\boldsymbol{W}_{hs} \in \mathbb{R}^{d_l \times d_l}$ are matrices of learnable weights. Then, the attentive weights are generated over the visual sentinel vector $\boldsymbol{sr}_t^v$ and the regions in $\boldsymbol{r}_t$:

$$[\boldsymbol{\alpha}_t^r; \alpha_t^v] = \text{softmax}([\hat{\boldsymbol{\alpha}}_t^r; \boldsymbol{w}_h^T \tanh(\boldsymbol{W}_{ss}\boldsymbol{sr}_t^v + \boldsymbol{W}_g\boldsymbol{h}_t^1)]), \tag{23}$$

where $\boldsymbol{W}_{ss} \in \mathbb{R}^{d_a \times d_l}$ is the learnable weights.

4. Merging Two Semantic Structures

The algorithm of merging two semantic structures (*i.e.*, sub-role sequences) is shown in Algorithm 1. Given multiple VSRs, we can continually use this algorithm by regarding the merged semantic structure as the first input structure.

Algorithm 1 Merging Algorithm of Semantic Structures

Input: Two semantic structures and corresponding sequence of grounded visual regions: $(\mathcal{S}^a, \mathcal{R}^a)$ and $(\mathcal{S}^b, \mathcal{R}^b)$.

Output: The merged semantic structure $\mathcal{S}$ and grounded visual regions $\mathcal{R}$.

$\mathcal{R} = \mathcal{R}^a$
// build a sequence of region sets $\mathcal{R}_{\text{same}}$, which is in both $\mathcal{R}^a$ and $\mathcal{R}^b$.
for each $\boldsymbol{r}_i^a \in \mathcal{R}^a$ **do**
  if $\boldsymbol{r}_i^a \in \mathcal{R}^b$ **then**
    $\mathcal{R}_{\text{same}}$.append($\boldsymbol{r}_i^a$)
  end if
end for
// if the rank of the same region sets in $\mathcal{R}^b$ is different from $\mathcal{R}^a$, re-rank those region sets.
$i_{\text{same}} = 0$
for each $\boldsymbol{r}_i^b \in \mathcal{R}^b$ **do**
  if $\boldsymbol{r}_i^b \in \mathcal{R}_{\text{same}}$ **then**
    $\boldsymbol{r}_i^b = \mathcal{R}_{\text{same}}[i_{\text{same}}]$
    i_{same} += 1
  end if
end for
// insert region sets in $\mathcal{R}^b \setminus R_{\text{same}}$ into $\mathcal{R}$.
for each $\boldsymbol{r}_i^b \in \mathcal{R}^b$ **do**
  if $\boldsymbol{r}_i^b \notin \mathcal{R}_{\text{same}}$ **then**
    insert $\boldsymbol{r}_i^b$ in $\mathcal{R}$ right before $\boldsymbol{r}_{\text{right}}^b$
    // $\boldsymbol{r}_{\text{right}}^b$ is the closest region set in the right of $\boldsymbol{r}_i^b$ in $\mathcal{R}^b$, which is also in $\mathcal{R}_{\text{same}}$.
  end if
end for
build $\mathcal{S}$ according to $\mathcal{R}$

5. Details of Experimental Settings

Parameter Settings. We use the Adam [2] optimizer in all our experiments. For the grounded semantic role labeling model, we initiate the learning rate to 1×10^{-5}, which decreases by a factor of 0.5 for every 3 epochs. To train the S-level SSP and R-level SSP, the learning rate is set to 1×10^{-4} and decreases by a factor of 0.6 for every 3 epochs. And the max training epoch is set to 20 for the models above. For the role-shift captioning model, the batch size is set to 100. The learning rate is 5×10^{-4} for XE training and 5×10^{-5} for the RL training, decreasing by a factor of 0.8 every epoch. The hidden size of both two LSTMs is set to 512. In the training stage, we apply early stopping according to the CIDEr-D score in the validation dataset. In the inference stage, we employ the beam search strategy with a beam size of 5.

Details of Training and Test. Due to the constraint of COCO/Flickr30k Entities, there are many captions containing nouns without region annotation. Thus, we followed [1] to fill the missing regions with most probable detections of the image in the training of role-shift caption model and drop these captions in validation and test stages. And those are also dropped in other models' training and test stages.

	Proposal	Model	B4	M	R	C	S
COCO	GSRL	SN	15.5	23.0	46.5	159.3	35.1
		TF	**16.0**	**23.2**	**47.1**	**162.8**	**35.7**
	GT	SN	22.3	27.6	54.2	227.9	48.1
		TF	**23.1**	**28.0**	**55.6**	**235.1**	**48.9**
Flickr30K	GSRL	SN	7.6	14.5	32.1	69.0	17.8
		TF	**7.9**	**14.7**	**32.6**	**71.6**	**18.2**
	GT	SN	9.6	17.3	35.4	86.9	21.2
		TF	**10.7**	**18.0**	**37.1**	**97.5**	**21.9**

Table 4. Performance comparisons between Transformer (TF) and Sinkhorn Network (SN) in S-level SSP on dataset COCO Entities and Flickr30K Entities.

6. Transformer vs. Sinkhorn Network in the S-level SSP.

Settings. To sort the sequence of roles from the given control signal, Sinkhorn network is another alternative network. To further compare the Transformer and Sinkhorn network in the S-level SSP, we design a strong baseline by replacing the Transformer to Sinkhorn network. The results on COCO Entities and Flickr30K Entities are reported in Table 4.

Results. From Table 4, we can observe that the model with Transformer can achieve better performance than the model with Sinkhorn network in all proposal settings (GSRL detected proposals or ground truth proposals) and evaluation metrics on both COCO Entities and Flickr30K Entities benchmarks. This may because that the Transformer can better encode the dependency on previous outputs (semantic roles). Thus, we use Transformer for our S-level SSP.

References

[1] Marcella Cornia, Lorenzo Baraldi, and Rita Cucchiara. Show, control and tell: A framework for generating controllable and grounded captions. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2019. 4

[2] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *arXiv*, 2014. 4

[3] Harold W Kuhn. The hungarian method for the assignment problem. *Naval research logistics quarterly*, 1955. 3

[4] Jiasen Lu, Caiming Xiong, Devi Parikh, and Richard Socher. Knowing when to look: Adaptive attention via a visual sentinel for image captioning. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2017. 3

[5] Gonzalo Mena, David Belanger, Scott Linderman, and Jasper Snoek. Learning latent permutations with gumbel-sinkhorn networks. In *Int. Conf. Learn. Represent.*, 2018. 3

[6] Martha Palmer, Daniel Gildea, and Paul Kingsbury. The Proposition Bank: An annotated corpus of semantic roles. *Computational Linguistics*, 2005. 1