Supplemental Material: Monolingual Phrase Alignment on Parse Forests

Yuki Arase[1⋆] and **Junichi Tsujii**[⋆2]
[1]Osaka University, Japan
[⋆]Artificial Intelligence Research Center (AIRC), AIST, Japan
[2]NaCTeM, School of Computer Science, University of Manchester, UK
arase@ist.osaka-u.ac.jp, j-tsujii@aist.go.jp

A Theorems and Proofs

This section provides a formal derivation of theorems in Sec. 3.2 and Sec. 3.3.

Since $l/ds(\cdot)$ and $r/ds(\cdot)$ in Definition 3.1 are sets, the same aligned pair may have more than one support of descendant alignments. Let us assume that an alignment $\mathbb{h}_i = \langle \tau_i^s, \tau_i^t \rangle$ is supported by more than one pair of descendant alignments in Δ_L. By the *Consistency* condition, all supports of a pair should belong to the same type (*i.e.*, either $\Rightarrow$ or $\overset{R}{\Rightarrow}$). Without a loss of generality, we can assume that all supports of a pair are $\Rightarrow$. That is, $\Delta_L \supseteq (\{\langle \mathbb{h}_m, \mathbb{h}_n \rangle\} \Rightarrow \mathbb{h}_i)$, where $\mathbb{h}_m = \langle \tau_m^s, \tau_m^t \rangle$ and $\mathbb{h}_n = \langle \tau_n^s, \tau_n^t \rangle$. Since all supports of $\mathbb{h}_i$ are $\Rightarrow$, $\tau_m^s \in l/ds(\tau_i^s) \wedge \tau_m^t \in l/ds(\tau_i^t)$ and $\tau_n^s \in r/ds(\tau_i^s) \wedge \tau_n^t \in r/ds(\tau_i^t)$ are satisfied. Let us denote $\mathbb{H}_m = \{\mathbb{h}_m\}$ and $\mathbb{H}_n = \{\mathbb{h}_n\}$. In the following, we use $\Rightarrow$ for the two types of support ($\Rightarrow$ and $\overset{R}{\Rightarrow}$).

The *Monotonous* and *Maximum Set* conditions allow Δ_L to be further restricted so that each of aligned pairs in $\mathbb{H}_L$ has only one support. Theorem 3.1 shows the existence of the maximum pair that satisfies:

Lemma A.1. $\langle \mathbb{h}_M, \mathbb{h}_N \rangle \Rightarrow \mathbb{h}_i$ *is in* Δ_L.

For each $\mathbb{h}_m \in \mathbb{H}_m$ and $\mathbb{h}_n \in \mathbb{H}_n$, if all support relations from Δ_L are removed except for the ones by the maximum pairs or the pre-terminal alignments, the resultant set Δ'_L satisfies:

Lemma A.2. $\{\mathbb{h}_p \overset{*}{\mapsto} \mathbb{h}_q\} \in \Delta_L \leftrightarrow \{\mathbb{h}_p \overset{*}{\mapsto} \mathbb{h}_q\} \in \Delta'_L$.

In Δ'_L, each aligned pair in $\mathbb{H}_L$ has only one support. Lemma A.2 implies that Δ'_L preserves the relationship of $\overset{*}{\mapsto}$ among the aligned pairs in Δ_L. Therefore, removing the other support relations does not affect the set of aligned pairs, $\mathbb{H}_L$. With these lemmas, Theorem 3.2 can be derived.

Below we prove the theorems and lemmas in order of their logical relations. First, Theorem 3.1 is proved as follows.

Proof. Let us assume $\exists \langle l(\tau_i^s), \tau^t \rangle \in \mathbb{H}_m$. $\tau_m^s \in l/ds(\tau_i^s)$ for $\forall \langle \tau_m^s, \tau_m^t \rangle \in \mathbb{H}_m$. Thus, $\forall \langle \tau_m^s, \tau_m^t \rangle \in \mathbb{H}_m, \tau_m^t \in ds(\tau^t)$ (*Monotonous* condition). This means $\mathbb{h}_M = \langle l(\tau_i^s), \tau^t \rangle$. If $\nexists \langle l(\tau_i^s), \cdot \rangle \in \mathbb{H}_m$, either $\forall \langle \tau_m^s, \tau_m^t \rangle \in \mathbb{H}_m, \tau_m^s \in l/l/ds(\tau_i^s)$ or $\forall \langle \tau_m^s, \tau_m^t \rangle \in \mathbb{H}_m, \tau_m^s \in l/r/ds(\tau_i^s)$ should be satisfied. Otherwise, there would be a pair of alignments that support $\langle l(\tau_i^s), \cdot \rangle$, and by the condition of the *Maximum set*, the pair should be in $\mathbb{H}_m$. Without a loss of generality, we can assume $\forall \langle \tau_m^s, \tau_m^t \rangle \in \mathbb{H}_m, \tau_m^s \in l/l/ds(\tau_i^s)$. Then we can repeat the above argument; if $\exists \langle l/l(\tau_i^s), \tau^t \rangle \in \mathbb{H}_m, \mathbb{h}_M = \langle l/l(\tau_i^s), \tau^t \rangle$. Otherwise, either $\forall \langle \tau_m^s, \tau_m^t \rangle \in \mathbb{H}_m, \tau_m^s \in l/l/l/ds(\tau_i^s)$ or $\forall \langle \tau_m^s, \tau_m^t \rangle \in \mathbb{H}_m, \tau_m^s \in l/l/r/ds(\tau_i^s)$. Since the process will trace a tree downward, it terminates at the pre-terminals. □

Lemma A.1 is obvious. In Lemma A.2, the sufficient condition is due to the definition. The necessary condition can be proved as follows.

Proof. $\{\mathbb{h}_p \overset{*}{\mapsto} \mathbb{h}_q\} \in \Delta_L$ can be expanded as $(\langle \mathbb{h}_p, \cdot \rangle \Rightarrow \mathbb{h}_{p+1}), \ldots, (\langle \mathbb{h}_j, \mathbb{h}_k \rangle \Rightarrow \mathbb{h}_l), \ldots, (\langle \mathbb{h}_{q-1}, \cdot \rangle \Rightarrow \mathbb{h}_q)$. For each $\langle \mathbb{h}_j, \mathbb{h}_k \rangle \Rightarrow \mathbb{h}_l$, there exist the maximum pairs $\langle \mathbb{h}_J, \mathbb{h}_K \rangle \Rightarrow \mathbb{h}_l \in \Delta_L$ where $\mathbb{h}_j \leq \mathbb{h}_J$ and $\mathbb{h}_k \leq \mathbb{h}_K$ (Lemma A.1). Δ'_L contains all maximum pairs, thus $\langle \mathbb{h}_J, \mathbb{h}_K \rangle \Rightarrow \mathbb{h}_l \in \Delta'_L$. Since $\mathbb{h}_j \overset{*}{\mapsto} \mathbb{h}_J$ and $\mathbb{h}_k \overset{*}{\mapsto} \mathbb{h}_K$ (*Same-Tree* condition), the chain relationship is retained in Δ'_L. □

Theorem 3.2 is obvious from the definition of Δ'_L and Lemma A.2.

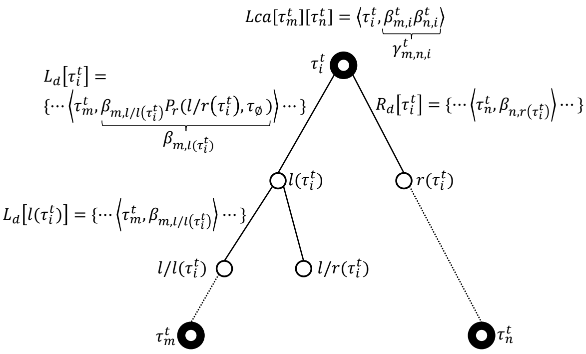

Figure 1: Bottom-up computation of $Lca[\tau_m][\tau_n]$

B Pseudo-code of Phrase Alignment

Algorithm B.1 depicts the pseudo-code of our alignment algorithm, which uses pre-computed $L_d^t[\cdot]$, $R_d^t[\cdot]$, and $Lca^t[\cdot][\cdot]$ for the target-side tree. On the other hand, $L_d^s[\cdot]$, $R_d^s[\cdot]$, and $Lca^s[\cdot][\cdot]$ for the source-side tree are computed on the fly for efficiency.

$Lca^t[\tau_m^t][\tau_n^t]$ stores a tuple $\langle \tau_i^t, \gamma_{m,n,i}^t \rangle$ where $\tau_i^t = lca(\tau_m^t, \tau_n^t)$[1]. $L_d^t[\tau_i^t]$ maintains a set of tuples of $\langle \tau_m^t, \beta_{m,i}^t \rangle$, which means that τ_m^t is the left-descendant of τ_i^t and that the path from τ_m^t to τ_i^t via $l(\tau_i^t)$ has $\beta_{m,i}^t$ as the probability. $R_d^t[\tau_i^t]$ stores the same information for the right-descendants. When τ_m^t and τ_n^t are a left and right descendant of τ_i^t, respectively, τ_i^t is the LCA of τ_m^t and τ_n^t with $\gamma_{m,n,i}^t = \beta_{m,i}^t \beta_{n,i}^t$. $L_d^t[\tau_i^t]$ and $R_d^t[\tau_i^t]$ can be computed easily from those of the child phrases, *i.e.*, $l(\tau_i^t)$ and $r(\tau_i^t)$, by tracing a tree in a bottom-up manner (Fig. 1).

Using these, Algorithm B.1 computes an array $A[\cdot]$ as well as $L_d^s[\cdot]$, $R_d^s[\cdot]$, and $Lca^s[\cdot][\cdot]$ for phrases in the source-side parse tree by tracing a tree in a bottom-up manner. It should be noted that only paths from descendant phrases already aligned are computed in line 14 to 20. Paths from non-aligned phrases do not contribute to the creation of new aligned pairs.

When τ_m^s and τ_n^s in $\mathbb{h}_m$ and $\mathbb{h}_n$ are the left and right descendants of τ_i^s (*i.e.*, $\langle \tau_m^s, \cdot \rangle$ and $\langle \tau_n^s, \cdot \rangle$ in $L_d^s[\tau_i^s]$ and $R_d^s[\tau_i^s]$, respectively), τ_i^s is the LCA of τ_m^s and τ_n^s to be aligned with the LCA of τ_m^t and τ_n^t. By retrieving τ_m^t and τ_n^t from $A[\tau_m^s]$ and $A[\tau_n^s]$, respectively, and their LCA (*i.e.*, τ_i^t) from $Lca^t[\tau_m^t][\tau_n^t]$, Algorithm B.1 creates $\mathbb{h}_i$, which is added to $A[\tau_i^s]$. Since both $A[\tau_m^s]$ and $A[\tau_n^s]$ are sets of competing aligned pairs, more than one τ_m^t and τ_n^t are generally retrieved. Because different pairs of τ_m^t and τ_n^t have their own LCA's (*i.e.*, τ_i^t), different alignments are constructed. The inside

[1] In the case of forests, we store the set of tuples because the same pair of phrases may have more than one LCA and the same LCA can be reached via more than one paths.

Algorithm B.1 Pseudo-Code of Phrase Alignment

1: set $A[\tau^s] \leftarrow \emptyset$ for all τ^s
2: **for all** $\langle w^s, w^t \rangle \in \mathbb{W}$ **do**
3: Find τ^s and τ^t covering w^s and w^t
4: $\alpha = P_r(\tau^s, \tau^t)$
5: PACK($\langle \tau^s, \tau^t \rangle, \langle \alpha, \emptyset \rangle, A$)
6: **for all** τ_i^s **do** ▷ Trace source tree from the bottom to the top
7: **if** τ_i^s is a pre-terminal phrase **then**
8: $L_d^s[\tau_i^s] \leftarrow \emptyset, R_d^s[\tau_i^s] \leftarrow \emptyset$
9: **else**
10: **if** $A[l(\tau_i^s)] \neq \emptyset$ **then**
11: $L_d^s[\tau_i^s] \leftarrow \langle l(\tau_i^s), 1 \rangle$
12: **else**
13: $L_d^s[\tau_i^s] \leftarrow \emptyset$
14: **for all** $\langle \tau_j^s, \beta_{j,l(\tau_i^s)}^s \rangle \in L_d^s[l(\tau_i^s)]$ **do**
15: $L_d^s[\tau_i^s] \leftarrow L_d^s[\tau_i^s] \cup$
16: $\langle \tau_j^s, \beta_{j,l(\tau_i^s)}^s P_r(l/r(\tau_i^s), \tau_\emptyset) \rangle$
17: **for all** $\langle \tau_j^s, \beta_{j,l(\tau_i^s)}^s \rangle \in R_d^s[l(\tau_i^s)]$ **do**
18: $L_d^s[\tau_i^s] \leftarrow L_d^s[\tau_i^s] \cup$
19: $\langle \tau_j^s, \beta_{j,l(\tau_i^s)}^s P_r(l/l(\tau_i^s), \tau_\emptyset) \rangle$
20: *Do equivalent process for $L_d^s[r(\tau_i^s)]$ and $R_d^s[r(\tau_i^s)]$ to compute $R_d^s[\tau_i^s]$
21: **for all** $\langle \tau_m^s, \beta_{m,i}^s \rangle \in L_d^s[\tau_i^s]$ **do**
22: **for all** $\langle \tau_n^s, \beta_{n,i}^s \rangle \in R_d^s[\tau_i^s]$ **do**
23: $Lca^s[\tau_m^s][\tau_n^s]$
24: $\leftarrow Lca^s[\tau_m^s][\tau_n^s] \cup \langle \tau_i^s, \beta_{m,i}^s \beta_{n,i}^s \rangle$
25: ALIGN($\tau_m^s, \tau_n^s, \tau_i^s, \beta_{m,i}^s \beta_{n,i}^s, A$)

26: **function** ALIGN($\tau_m^s, \tau_n^s, \tau_i^s, \gamma^s, A$)
27: **for all** $\mathbb{h}_m = \langle \tau_m^s, \tau_m^t \rangle \in A[\tau_m^s]$ **do**
28: **for all** $\mathbb{h}_n = \langle \tau_n^s, \tau_n^t \rangle \in A[\tau_n^s]$ **do**
29: $\langle \tau_i^t, \gamma^t \rangle \leftarrow Lca^t[\tau_m^t][\tau_n^t]$
30: $\alpha = \max_\alpha(\mathbb{h}_m) \max_\alpha(\mathbb{h}_n) \gamma^s \gamma^t P_r(\tau_i^s, \tau_i^t)$
31: PACK($\langle \tau_i^s, \tau_i^t \rangle, \langle \alpha, \langle \mathbb{h}_m, \mathbb{h}_n \rangle \rangle, A$)

32: **function** PACK($\langle \tau^s, \tau^t \rangle, \langle \alpha, \langle \mathbb{h}_m, \mathbb{h}_n \rangle \rangle, A$)
33: **if** $\langle \tau^s, \tau^t \rangle \in A[\tau^s]$ **then**
34: $A[\tau^s] \leftarrow A[\tau^s] \cup \langle \alpha, \langle \mathbb{h}_m, \mathbb{h}_n \rangle \rangle$ ▷ Merge supports and their inside probability
35: **else**
36: $A[\tau^s] \leftarrow (\langle \tau^s, \tau^t \rangle, \langle \alpha, \langle \mathbb{h}_m, \mathbb{h}_n \rangle \rangle)$

probability of each of these new pairs can be computed from α_m of $\mathbb{h}_m$ in $A[\tau_m^s]$ and α_n of $\mathbb{h}_n$ in $A[\tau_n^s]$ as well as $\gamma_{m,n,i}^s$ in $Lca^s[\tau_m^s][\tau_n^s]$ and $\gamma_{m,n,i}^t$ in $Lca^t[\tau_m^t][\tau_n^t]$ in Algorithm B.1, line 30. The function $\max_\alpha(\cdot)$ determines α_m and α_n as:

$$\max_{\alpha_j \in \{\langle \alpha_j, \langle \mathbb{h}_l, \mathbb{h}_r \rangle \rangle\}} \alpha_j.$$

C Pseudo-code for Non-compositional Alignment

Algorithm C.1 is the pseudo-code of the non-compositional alignment. The following notations are used: $[\tau_m]^i$ and $[\tau_n]^j$ represent the phrases of τ_m and τ_n with the i-th and j-th sets of supporting alignments, respectively. $\Psi^{[\tau_m]^i} = \{\psi_k^{[\tau_m]^i}\}, \Psi^{[\tau_n]^j} = \{\psi_k^{[\tau_n]^j}\}$ are the sets of aligned phrases in $[\tau_m]^i$ and $[\tau_n]^j$, respectively. $\Phi^{[\tau_m]^i} = \{\phi_l^{[\tau_m]^i}\}, \Phi^{[\tau_n]^j} = \{\phi_l^{[\tau_n]^j}\}$ are the sets of null-alignments in $[\tau_m]^i$ and $[\tau_n]^j$, respectively.

Algorithm C.1 takes two arguments, τ_m and τ_n, and checks whether there are supporting alignments by which $[\tau_m]^i$ and $[\tau_n]^j$ are compatible. The function returns a set of tuples $\{\langle \Psi^k, \Phi^k \rangle\}$. If the returned set is empty, τ_m and τ_n are incompatible. More precisely, their alignments with the source phrases are incompatible in the sense that no pair of their supporting alignments make their null-alignments and aligned phrases compatible. They fail to create a new non-monotonic alignment pair.

In Algorithm C.1, $Sp(\cdot)$ enumerates different supporting alignments. Let us consider that $Sp(\psi_l^{[\tau_m]^i}) = \{[\psi_l^{[\tau_m]^i}]^k\}$. $\psi_l^{[\tau_m]^i}$ is one of the target phrases inside τ_m, which is aligned with a phrase in the source by the i-th support set of alignment $\langle \cdot, \tau_m \rangle$. On the other hand, $[\psi_l^{[\tau_m]^i}]^k$ denotes the same phrase, but it has its own internal structure in terms of aligned phrases and null-alignments. The internal structure is determined by the k-th support set of $\langle \cdot, \psi_l^{[\tau_m]^i} \rangle$. Since a tuple of $\langle \Psi^{[\tau_n]^j}, \Phi^{[\tau_n]^j} \rangle$ determines the internal structure of $[\tau_n]^j$, we can treat a tuple of $\langle \Psi, \Phi \rangle$, where Ψ and Φ are sets of aligned phrases and null-alignments in τ_n, in the same way as $[\tau_n]^j$. The functions of DOWN and COMPATIBILITY in Algorithm C.1 use this property.

In Algorithm C.1, line 29, the MERGE function merges two sets of tuples. TRACE(τ_n, ψ) returns a set of tuples $\{\langle \Psi^l, \Phi^l \rangle\}$, where all phrases in Ψ^l and Φ^l are descendants of ψ. We can create a new set of tuples by merging $\langle \Psi^k, \Phi^k \rangle$ with them, that is, $\{\langle \Psi^l \cup (\Psi^k \setminus \{\psi\}), \Phi^l \cup \Phi^k \rangle\}$.

When TRACE(τ_m^t, τ_n^t) in Algorithm C.1 returns a set of $\{\langle \Psi^k, \Phi^k \rangle\}$, all $\psi_l \in \Psi^k$ are aligned with phrases in the source and their inside probabilities are stored in A . We can compute the inside probability for each $\langle \Psi^k, \Phi^k \rangle$. A new alignment pair $\langle \tau_i^s, \tau_i^t (= \tau_m^t) \rangle$ where $\tau_i^s = lca(\tau_m^s, \tau_n^s)$ and their supports $\{\langle \Psi^k, \Phi^k \rangle\}$ with their inside probabilities is stored in A.

Algorithm C.1 Pseudo-code of non-compositional alignment

1: **function** TRACE(τ_n, τ_m) $\quad \triangleright\ \tau_n \in ds(\tau_m)$
2: $\quad V \leftarrow \emptyset$
3: $\quad$ **if** τ_m is a pre-terminal phrase **then**
4: $\quad\quad$ **return** $\emptyset$;
5: $\quad$ **for all** $[\tau_m]^i \in Sp(\tau_m)$ **do**
6: $\quad\quad$ **if** $\tau_n \in ds(\phi)$ for $\exists \phi \in \Phi^{[\tau_m]^i}$ **then**
7: $\quad\quad\quad V \leftarrow V \cup \langle \Psi^{[\tau_m]^i} \cup \tau_n, (\Phi^{[\tau_m]^i} \setminus \phi) \cup$ GAP$(\tau_n, \phi) \rangle$
8: $\quad\quad$ **else if** $\tau_n \in ds(\psi)$ for $\exists \psi \in \Psi^{[\tau_m]^i}$ **then**
9: $\quad\quad\quad V \leftarrow V \cup$ TRACE(τ_n, ψ)
10: $\quad\quad$ **else**
11: $\quad\quad\quad$ **for all** $[\tau_n]^j$ **do**
12: $\quad\quad\quad\quad V \leftarrow V \cup$ DOWN$([\tau_n]^j, [\tau_m]^i)$
13: $\quad$ **return** V;

14: **function** DOWN$([\tau_n]^j, [\tau_m]^i)$
15: $\quad V \leftarrow [\tau_m]^i$
16: $\quad$ **for all** $\psi_l \in \Psi^{[\tau_n]^j}$ **do**
17: $\quad\quad V \leftarrow$ COMPATIBILITY(ψ_l, V)
18: $\quad\quad$ **if** V is empty **then**
19: $\quad\quad\quad$ **return** $\emptyset$
20: $\quad$ **return** V;

21: **function** COMPATIBILITY(τ_n, C)
22: $\quad V \leftarrow \emptyset$
23: $\quad$ **for all** $\langle \Psi^k, \Phi^k \rangle \in C$ **do**
24: $\quad\quad$ **if** $\tau_n \in ds(\phi)$ for $\exists \phi \in \Phi^k$ **then**
25: $\quad\quad\quad V \leftarrow V \cup \langle \Psi^k \cup \tau_n, (\Phi^k \setminus \phi) \cup$ GAP$(\tau_n, \phi) \rangle$
26: $\quad\quad$ **else if** $\tau_n \in ds(\psi)$ for $\exists \psi \in \Psi^k$ **then**
27: $\quad\quad\quad V' \leftarrow$ TRACE(τ_n, ψ)
28: $\quad\quad\quad$ **if** $V' \neq \emptyset$ **then**
29: $\quad\quad\quad\quad V \leftarrow V \cup$ MERGE$(\langle \Psi^k, \Phi^k \rangle, V')$
30: $\quad\quad$ **else**
31: $\quad\quad\quad$ **for all** $[\tau_n]^j$ **do**
32: $\quad\quad\quad\quad V \leftarrow V \cup$ DOWN$([\tau_n]^j, \langle \Psi^k, \Phi^k \rangle)$
33: $\quad$ **return** V

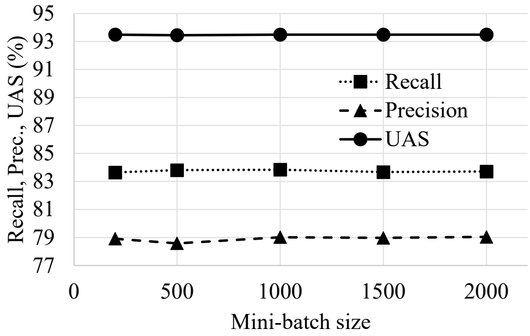

Figure 2: Effect of mini-batch size on EM training

D Evaluation Details

D.1 Detailed Statistics on Results

Table 1 shows results on the development and test sets with p-values in significance testing. The significance test is conducted by comparing each method to the proposed method.

D.2 Effect of Forest Size

We investigated the effect of the size of parse forests. To obtain forests of a larger size, the parameters in Enju (Ninomiya et al., 2005) are changed, increasing the average number of nodes in a forest of the entire training corpus from 339 to 520. The hyper-parameters are set to the ones with the best performance in the development set as shown in Table 1: $\mu_n = 1.0, \mu_c = 3.0, \mu_p = 0.7, \mu_b = 150, \mu_g = 5$ ($\mu_b = 50$ during EM training). EM is conducted with the entire training corpus with a mini-batch size of 500 due to memory consumption using larger forests.

Consequently, the recall and precision of the alignment quality are 84.06% and 79.25%, while UAS is 93.34%, where a significant difference is not observed compared to the model trained on the previous set of (smaller) forests using the same mini-batch size of 500 (p-values are 0.07, 0.27, and 1.00 for recall, precision, and UAS, respectively). Larger forests show more potential to improve the recall of the alignment quality, a larger μ_b may be necessary to effectively make use of such large forests where more alignment candidates are observed.

D.3 Effect of Mini-Batch Size

We also investigated the effect of the mini-batch size in EM training using the entire training corpus ($41K$ pairs). The hyper-parameters were set to $\mu_n = 1.0, \mu_c = 3.0, \mu_p = 0.7, \mu_b = 150, \mu_g = 5$ (again, $\mu_b = 50$ during EM training).

Fig. 2 shows the recall, precision, and UAS as functions of the mini-batch size. They are fairly stable against not only the mini-batch size but also the amount of training corpus (recall that the model using mini-batch size of 200 is trained on $2K$ samples). This demonstrates that our method can be trained with a moderate amount of data.

D.4 Example Alignments

Table 2 and Table 3 show the phrase alignment results by our method, where near-duplicate alignments due to the hierarchy in phrase structures (*e.g.*, alignments of parent and child phrases with only a single token difference) are omitted for clarity. Table 2 uses a simpler example where monotonic phrase alignment works, while the one in Table 3 requires non-compositional alignment to align divergent structures in source and target.

References

Takashi Ninomiya, Yoshimasa Tsuruoka, Yusuke Miyao, and Jun'ichi Tsujii. 2005. Efficacy of beam thresholding, unification filtering and hybrid parsing in probabilistic HPSG parsing. In *Proceedings of the International Workshop on Parsing Technology (IWPT)*, pages 103–114, Vancouver, British Columbia.

Method	UAS (Dev)	Recall	Prec.	UAS (Test)
Human	–	90.65	88.21	–
Proposed	92.79	**83.64**	**78.91**	93.49
Monotonic	93.04	82.86*$(p = 0.01)$	77.97*$(p = 0.03)$	93.49
w/o EM	93.17	81.33*$(p = 0.02)$	75.09*$(p = 0.01)$	92.91*$(p = 0.00)$
1-best tree	–	80.11*$(p = 0.00)$	73.26*$(p = 0.00)$	93.56 $(p = 1.00)$

Table 1: Evaluation results on development and test sets with p-values in significance testing

Source	Target
The four female doctors in the team have become the first Chinese women aid workers to carry out a mission outside the country	The four female doctors in the team have become China 's first female rescue workers to carry out a mission overseas
The four female doctors in the team	The four female doctors in the team
have become the first Chinese women aid workers to carry out a mission outside the country	have become China 's first female rescue workers to carry out a mission overseas
become the first Chinese women aid workers	become China 's first female rescue workers
the first Chinese women aid workers	China 's first female rescue workers
women aid workers	female rescue workers
to carry out a mission outside the country	to carry out a mission overseas
a mission outside the country	a mission overseas
outside the country	overseas

Table 2: Example of monotonic phrase alignments

Source	Target
The 26-year AkshayVishal of Secunderabad was shot two days ago in Arkansas by unidentified persons	An unidentified assailant shot 26-year-old Akshay Vishal of Secunderabad two days ago in the state of Arkansas
The 26-year AkshayVishal of Secunderabad	26-year-old Akshay Vishal of Secunderabad
was shot two days ago in Arkansas by unidentified persons	shot 26-year-old Akshay Vishal of Secunderabad two days ago in the state of Arkansas
shot two days ago	shot 26-year-old Akshay Vishal of Secunderabad two days ago
in Arkansas by unidentified persons	in the state of Arkansas

Table 3: Example of phrase alignments with non-compositional alignment