Supplementary Material

Chunchuan Lyu[1] **Ivan Titov**[1,2]
[1]ILCC, School of Informatics, University of Edinburgh
[2]ILLC, University of Amsterdam

1 Matching algorithm for copying concepts

Only frequent concepts c (frequency at least 10 for R2 and 5 for R1) can be generated without the copying mechanism (i.e. have their own vector $\mathbf{v}_c$ associated with them). Both frequent and infrequent ones are processed with coping, using candidates produced by the algorithm below and the matching rule in Table 1.

```
Input  : {w^l, c^l}_{l=1}^L
Output: D copy dictionary
Counter ← ∅
for l = 1 to L do
    for all pairs c_i^l and w_j^l do
        if match(c_i^l, w_j^l) then
            Increment Counter[w_j^l][c_i^l]
        end
    end
end
D ← default Stanford lemmatizer
for w ← Counter do
    D[w] ← argmax_c Counter[w][c]
end
return D
```

Algorithm 1: Copy function construction

Rules	Matching Criteria
Verbalization Match	exact match frame in ”verbalization-list-v1.06.txt”
PropBank Match	exact match frame in PropBank frame files
Suffix Removal Match	word with suffix (“-ed”, “-ly”,“-ing”) removed is identical to concept lemma
Edit-distance Match	edit distance smaller than 50% of the length

Table 1: Matching rules for Algorithm 1

2 Re-categorization details

Re-categorization is handled with rules listed in Table 2. They are triggered if a given primary concept (‘primary’) appears adjacent to edges labeled with relations given in column ‘rel’. The assigned category is shown in column ‘re-categorized’. The rules yield 32 categories when applied to the training set.

There are also rules of another type shown in Table 3 below. The templates and examples are in column

primary	rel	re-categorized
person	ARG0-of/ARG1-of	person([second])
thing	ARG0-of/ARG1-of/ARG2-of	thing([second])
most	degree-of	most([second])
-quantity	unit	primary([second])
date-entity	weekday/dayperiod/season	date-entity([second])
monetary-quantity	unit/ARG2-of/ARG1-of/quant	monetary-quantity([second])
temporal-quantity	unit/ARG3-of	temporal-quantity([second])

Table 2: Templates for re-categorization.

'original', the resulting concepts are in column 're-categorized'. These rules yield 109 additional types when applied to the training set.

original	re-categorized
(c / type :name (n / name :op1 'n1' ... :opx 'nx')	(B-Ner_type(n1),...,Ner_type(nx))
(c / city :name (n / name :op1 'New' :op2 'York')	B-Ner_city(New),Ner_city(York)
(p / type :ARG0-of (h / have-x-role-91 :ARG2 (p / role)	have-x-role_type(role)
(p / person :ARG0-of (h / have-org-role-91 :ARG2 (p / premier)	have-org-role_person(premier)
(o1 / x-entity :x constant)	x-entity(constant)
(o1 / ordinal-entity :value 1)	ordinal-entity(1)

Table 3: Extra rules for re-categorization.

3 Additional pre-processing

Besides constructing re-categorized AMR concepts, we perform additional preprocessing. We start with tokenized dataset of Pourdamghani et al. (2014). We take all dashed AMR concepts (e.g, *make-up* and *more-than*) and concatenate the corresponding spans (based on statistics from training set and PropBank frame files). We also combine spans of words corresponding to a single number. For relation identification, we normalize relations to one canonical direction (e.g. arg0, time-of). For named entity recognition, and lemmatization, we use Stanford CoreNLP toolkit (Manning et al., 2014). For pre-trained embedding, we used Glove (300 dimensional embeddings) (Pennington et al., 2014).

4 Model parameters and optimization details

We selected hyper-parameters based on the best performance on the development set. For all the ablation tests, the hyper parameters are fixed. We used 2 different BiLSTM encoders of the same hyper-parameters to encode sentence for concept identification and alignment prediction, another BiLSTM to encode AMR concept sequence for alignment, and finally 2 different BiLSTM of the same hyper-parameters to encode sentence for relation identification and root identification. There are 5 BiLSTM encoders in total. Hyper parameters for the model are summarized in Table 4, and optimization parameters are summarized in Table 5.

Model components	Hyper-parameters
Glove Embeddings	300
Lemma Embeddings	200
POS Embeddings	32
NER Embeddings	16
Category Embeddings	32
Concept/Alignment	1 layer 548 input
Sentence BiLSTM	256 hidden (each direction)
AMR Categories $\mathcal{T}$	32
AMR Lemmas $\mathcal{C}$	506
AMR NER types	109
Alignment	1 layer 232 input
AMR BiLSTM	100 hidden (each direction)
B bilinear align	200×512
Relation map dimensionality d_g	200
Relation/Root	2 layers 549 input (predicate position)
Sentence BiLSTM	256 hidden (each direction)
d_f relation vector	200
v_c, v_{copy} lemma vector	512
v_{root} root vector	200
Sinkhorn temperature	1
Sinkhorn prior temperature	5
Sinkhorn steps l for full joint training	10
Sinkhorn steps l for two stages training	5
λ	10
Dropout	.2

Table 4: Model hyper-parameters

Optimizer Parameters	Values
Batch size for single stage	64
Maximum Epochs	30
Batch size for first stage	512
Batch size for second stage	64
Maximum Epochs for both stages	30
Learning Rate	1e-4
Adam betas	(0.9, 0.999)
Adam eps	1e-8
Weight decay	1e-5

Table 5: Optimization parameters for full joint training and two stages training.

References

Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *Association for Computational Linguistics (ACL) System Demonstrations*, pages 55–60.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Nima Pourdamghani, Yang Gao, Ulf Hermjakob, and Kevin Knight. 2014. Aligning english strings with abstract meaning representation graphs. In *EMNLP*.