One SPRING to Rule Them Both: Symmetric AMR Semantic Parsing and Generation without a Complex Pipeline Supplementary Material

1 Postprocessing

Algorithm 1 Restore parenthesis parity

```
Input: D, the decoder output tokens
Output: O, the tokens with parentheses restored
C := 0
for i := 0 to len(D) − 1 do
   t := D_i
   if t = ")" then
      c := c + 1
   else if t = ")" then
      c := c − 1
   end if
   append token to O
   if c = 0 then
      break
   end if
end for
// if there are more ) than (
while c > 0 do
   append token to O
   c := c − 1
end while
```

We use a very light postprocessing approach. Our post-processing code's aim is recovering a valid graph from a possibly invalid output. Reasons why the output might be invalid include:

- parentheses do not match (only relevant for PENMAN and DFS);
- there are more than one `:instance` attributes for some variable;
- there is an invalid pair of tokens, such as two subsequent edge labels.

If parentheses do not match, we restore parity by using the simple approach described in Algorithm 1, which entails either removing all tokens to the right of the rightmost closed parenthesis, or adding appending right parenthesis until all open parentheses are matched. In the case of duplicate `:instance` attributes, we only keep the first one.

Parameter	Pick	Search Space
Training		
Loss	Cross-entr.	-
Optimizer	RAdam	-
W. Decay	0.004	0.001 to 0.01, (+0.001)
Betas	0.9, 0.999	-
Hardware	1080 Ti	-
Batch size	500	-
Grad. accum.	10	1/5/10/15/20
LR	$5 * 10^{-5}$	1/5/10/50 $*10^{-5}*$
LR sched.	constant	-
Epochs	30	-
Dropout	0.25	0.1 to 0.25, (0.05)
Prediction		
Beam size	5	[1,5]

Table 1: Hyperparameters and search space

As regards invalid subsequent token pairs, we scan the sequence of decoded tokens $D = t_0, \ldots, t_n$ from left to right, and if we find a couple t_i, t_{i+1} such that t_{i+1} cannot possibly follow t_i in a valid linearization, we discard t_{i+1}. After the removal, we go on discarding token $t_{i+2}, t_{i+3}, \ldots$ until we find one that can occur after t_i. After finding it, we continue scanning the sequence from the rightmost considered token.

2 Reproducibility details

2.1 SPRING Hyperparameters

We report in table 1 the final hyperparameters used to train and evaluate both the Text-to-AMR and AMR-to-Text models. To pick these parameters, we used random search with about 25 Text-to-AMR trials in the search space indicated in the third column of Table 1. Text-to-AMR training requires about 22 and 30 hours on, respectively AMR 2.0 and AMR 3.0; in contrast, AMR-to-Text only requires 13 and 16.5 hours on the same setting. We picked a beam size of 5 on the basis of common practice in Neural Machine Translation (Yang, Huang, and Ma 2018). Higher values showed only slight gains on the development sets.

Model	Smatch	Wikification
SPRING$^{\mathrm{DFS}}$	<u>**83.8**</u>	<u>**84.3**</u>
SPRING$^{\mathrm{BFS}}$	83.2	83.5
SPRING$^{\mathrm{PM}}$	83.6	83.1
SPRING$^{\mathrm{DFS}}$ -blink	<u>83.5</u>	79.3
SPRING$^{\mathrm{BFS}}$ -blink	83.0	<u>79.8</u>
SPRING$^{\mathrm{PM}}$ -blink	83.3	76.3

Table 2: Text-to-AMR parsing results (AMR 2.0). Row blocks: SPRING linerization variants (included in main paper), SPRING linearization variants without BLINK wikification; Best result per measure across table/row block in **bold**/<u>underlined</u>.

2.2 Vocabulary details

We include frames occurring more than 5 times in the training set, for a total of 2523 token types. Most amr-specific concepts (such as `date-entity`, `amr-unknown`), including the special reifications (e.g. `be-located-at-91`) are added to the vocabulary. In general, all of the relation labels are included. However, for what concerns enumerative labels, i.e. :snt$n$ and :op$n$, which are used to introduce, respectively, sentence subgraphs, and other kinds of coordinates subgraphs (e.g. units linked by *and*), we include them up to $n = 5$.

3 BLINK Wikification

In this Section we show additional results to those included in the main paper, to justify the usage of the off-the-shelf Entity linker, i.e., BLINK (Wu et al. 2019), in the postprocessing pipeline. In Table 2 we report the performance of SPRING variants (as shown in main results of the paper) and their performance when we do not employ BLINK (-blink). As one can see, BLINK is not necessary for achieving significantly higher overall Smatch score, as it provides a maximum of 0.3 Smatch improvement with SPRING$^{\mathrm{DFS}}$, thus suggesting that we can further simplify the postprocessing of our models and still achieve state-of-the-art results. However, it ensures a higher F1 performance in the Wikification metric, with an improvement ranging from 3.2 to 5 points with SPRING$^{\mathrm{PM}}$ and SPRING$^{\mathrm{DFS}}$, respectively.

4 Parsing examples

In this last section we report a few examples of parsing and generation obtained by running our DFS-based models trained on AMR 2.0. We collect some a few excerpts from the prompts shown by Radford et al. (2019), parse them into graphs and generate a sentence from the parsed graph. Results are shown in Table 3. We also include them (with the graph linearization indented for better readability) in the `samples.txt` file in the provided code. As one can see, the generated sentenced from the parsed graphs preserve the meaning of the original sentence, thus demonstrating the high quality of the outputs from both SPRING$^{\mathrm{DFS}}$ parser and generator. Note that the sentences are quite diverse, including things that are probably not present in the training data of SPRING$^{\mathrm{DFS}}$ – thus confirming its generalizability power.

References

Radford, A.; Wu, J.; Child, R.; Luan, D.; Amodei, D.; and Sutskever, I. 2019. Language Models are Unsupervised Multitask Learners. - .

Wu, L.; Petroni, F.; Josifoski, M.; Riedel, S.; and Zettlemoyer, L. 2019. Zero-shot Entity Linking with Dense Entity Retrieval. *CoRR* abs/1911.03814.

Yang, Y.; Huang, L.; and Ma, M. 2018. Breaking the Beam Search Curse: A Study of (Re-)Scoring Methods and Stopping Criteria for Neural Machine Translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 3054–3059. Brussels, Belgium: Association for Computational Linguistics. doi:10.18653/v1/D18-1342. URL https://www.aclweb.org/anthology/D18-1342.

Original	**→ Parsed graph**	**→ Generated Sentence**
In a shocking finding, scientist discovered a herd of unicorns living in a remote, previously unexplored valley, in the Andes Mountains.	`(z0 / discover-01 :ARG0 (z1 / scientist) :ARG1 (z2 / herd :consist-of (z3 / unicorn) :ARG0-of (z4 / live-01 :location (z5 / valley : mod (z6 / remote) :ARG1-of (z7 / explore-01 :polarity - :time (z8 / previous)) : location (z9 / mountain :wiki "Andes" :name (z10 / name :op1 "Andes" :op2 "Mountains"))))) :ARG0-of (z11 / shock-01))`	Scientists were shocked to discover a herd of unicorns living in a remote valley inaccessible in the Andes Mountains.
Emily loves mint chocolate cake, but she requires that it be paired with mini chocolate chips, so I threw some of those in between the layers.	`(z0 / love-01 :ARG0 (z1 / person :wiki - :name (z2 / name :op1 "Emily")) :ARG1 (z3 / cake : consist-of (z4 / chocolate :mod (z5 / mint))) :concession-of (z6 / require-01 :ARG0 z1 :ARG1 (z7 / pair-01 :ARG1 z3 :ARG2 (z8 / chip :consist-of (z9 / chocolate :mod (z10 / mini))))) :ARG0-of (z11 / cause-01 :ARG1 (z12 / throw-01 :ARG0 (z13 / i) :ARG1 (z14 / some :ARG1-of (z15 / include-91 :ARG2 z3)) :ARG2 (z16 / between :op1 (z17 / layer)))))`	Emily loves chocolate cake, but it requires it to be paired with mini chocolate chips, so I threw some of them in between the layers.
Prehistoric man sketched an incredible array of prehistoric beasts on the rough limestone walls of a cave in modern day France 36,000 years ago.	`(z0 / draw-01 :ARG0 (z1 / man :mod (z2 / prehistoric)) :ARG1 (z3 / array :mod (z4 / incredible) :consist-of (z5 / beast :mod (z6 / prehistoric))) :location (z7 / wall : consist-of (z8 / limestone) :ARG1-of (z9 / rough-04) :part-of (z10 / cave :location (z11 / country :wiki "France" :name (z12 / name :op1 "France") :time (z13 / day :ARG1-of (z14 / modern-02))))) :time (z15 / before :op1 (z16 / now) :quant (z17 / temporal-quantity :quant 36000 :unit (z18 / year))))`	36,000 years ago, prehistoric men drew an incredible array of prehistoric beasts on a rough limestone wall of a cave in modern-day France.
Corporal Michael P. Goeldin was an unskilled laborer from Ireland when he enlisted in Company A in November 1860.	`(z0 / person :ARG0-of (z1 / labor-01 :manner (z2 / skill :polarity -)) :domain (z3 / person :wiki - :name (z4 / name :op1 "Michael" : op2 "P." :op3 "Goeldin") :ARG0-of (z5 / have-org-role-91 :ARG2 (z6 / corporal))) : mod (z7 / country :wiki "Ireland" :name (z8 / name :op1 "Ireland")) :time (z9 / enlist -01 :ARG1 z3 :ARG2 (z10 / military :wiki - :name (z11 / name :op1 "Company" :op2 "A")) :time (z12 / date-entity :year 1860 :month 11)))`	When Michael P. Goeldin enlisted in Company A in November, 1860, he was an Irish labourer with no skills.
This pairing was the first outfit I thought of when I bought the shoes.	`(z0 / outfit :ord (z1 / ordinal-entity :value 1) :ARG1-of (z2 / think-01 :ARG0 (z3 / i) : time (z4 / buy-01 :ARG0 z3 :ARG1 (z5 / shoe ))) :domain (z6 / pair-01 :mod (z7 / this)) )`	This pair is the first outfit I thought of when I bought shoes.
The pink ghost's AI is designed to "feel" opposite of the red ghost's behavior.	`(z0 / design-01 :ARG1 (z1 / system :mod (z2 / artificial) :poss (z3 / ghost :ARG1-of (z4 / pink-04))) :ARG3 (z5 / feel-01 :ARG0 z1 : ARG1 (z6 / opposite-01 :ARG2 (z7 / behave -01 :ARG0 (z8 / ghost :mod (z9 / red))))))`	The artificial system of the pink ghosts was designed to feel the opposite of the way the red ghosts behaved.
Xresources can be an absolute pain (they were for me).	`(z0 / possible-01 :ARG1 (z1 / pain-01 :ARG0 (z2 / resource :mod (z3 / xresources)) :mod (z4 / absolute) :ARG1-of (z5 / cause-01 :ARG0 (z6 / they :beneficiary (z7 / i)))))`	The x-resoures could absolutely cause pain to me.

Table 3: Sentences from prompts used in the GPT-2 paper (Radford et al. 2019). Original sentence (left column); graph parsed with the DFS-based Text-to-AMR model trained on AMR 2.0 (middle column); sentence generated with the DFS-based AMR-to-Text model trained on AMR 2.0 from the parsed graph (right column).