Multitask Parsing Across Semantic Representations
Supplementary Notes

Features. Table 1 lists all feature used for the classifier (see §4.2). Numeric features are taken as they are, whereas categorical features are mapped to real-valued embedding vectors. For `w` features, we concatenate randomly-initialized and pre-trained word embeddings. For each node, we select a *head terminal* by traversing the graph according to a priority order on edge labels, taken from Hershcovich et al. (2017).

s_i refers to stack node i from the top, and b_i to buffer node i. xl and xr refer to a x's leftmost and rightmost children, and xL and xR to its leftmost and rightmost parents.

`w` refers to the node's head terminal text, `t` to its POS tag, and `d` to its dependency relation. `h` refers to the node's height, `e` to the tag of its first incoming edge, `n` and `c` to the node label and category (used only for AMR), `p` to any separator punctuation between s_0 and s_1, `q` to the count of any separator punctuation between s_0 and s_1, `x` to the numeric value of gap type (Maier and Lichte, 2016), `y` to the sum of gap lengths, `P`, `C`, `I`, `E`, and `M` to the number of parents, children, implicit children, remote children, and remote parents, `N` to the numeric value of the head terminal's named entity IOB indicator, `T` to its named entity type, `#` to its word shape (capturing orthographic features, e.g. ”Xxxx” or ”dd”), `^` to its one-character prefix, and `$` to its three-character suffix.

$x \rightarrow y$ refers to the existing edge from x to y. `x` is an indicator feature, taking the value of 1 if the edge exists or 0 otherwise, `e` refers to the edge label, and `d` to the dependency distance between the head terminals of the nodes.

a_i to the transition taken $i+1$ steps ago. `A` refers to the action type label (e.g. SHIFT/RIGHT-EDGE/NODE), and `e` to the edge label created by the action (e.g. $C/E/P$).

`node ratio` is the ratio between non-terminals and terminals, taken from Hershcovich et al. (2017).

Nodes	**Features**
s_0	`wtdencpT#^$xhqyPCIEMN`
s_1	`wtdencT#^$xhyN`
s_2	`wtdencT#^$xhy`
s_3	`wtdencT#^$xhyN`
b_0	`wtdncT#^$hPCIEMN`
b_1, b_2, b_3	`wtdncT#^$`
$s_0l, s_0r, s_1l, s_1r, s_0ll, s_0lr, s_0rl, s_0rr, s_1ll, s_1lr, s_1rl, s_1rr$	`wenc#^$`
$s_0L, s_0R, s_1L, s_1R, b_0L, b_0R$	`wen#^$`
Edges	
$s_0 \rightarrow s_1, s_0 \rightarrow b_0$	`xd`
$s_1 \rightarrow s_0, b_0 \rightarrow s_0$	`x`
$s_0 \rightarrow b_0, b_0 \rightarrow s_0$	`e`
Past actions	
a_0, a_1	`eA`
Misc.	`node ratio`

Table 1: Transition classifier features.

Conversion to and from Unified DAG Format. Although all experiments reported in the paper with the auxiliary tasks (AMR, DM and UD) are using unlabeled parsing for these schemes, our conversion code supports full conversion to and from these formats, and is publicly available at `http://github.com/danielhers/semstr/tree/master/semstr/conversion`.

Conversion from AMR to the unifid DAG format and back results in 95% Smatch F_1 (Cai and Knight, 2013) when averaged over the LDC2017T10 test set. On SDP, the conversion is lossless and results in identical graphs when converted to UCCA and back. For UD, and conversion results in 98.5% LAS F_1 on the UD English test set, due to multi-word tokens, not supported in the unified DAG format.

Qualitative evaluation. Figure 1 shows an example sentence from the English 20K test set, with the outputs of both our single-task model and our best MTL model (using all auxiliaries). While the single-task model obtains an F_1 score of 67.9% on this sentence, the MTL model's output matches the gold-annotates graph perfectly. This example demonstrates how the parser's ability to identify syntactic constituents, which is important for all tasks we tackled, is improved with MTL.

References

Shu Cai and Kevin Knight. 2013. Smatch: an evaluation metric for semantic feature structures. In *Proc. of ACL*, pages 748–752.

Daniel Hershcovich, Omri Abend, and Ari Rappoport. 2017. A transition-based directed acyclic graph parser for UCCA. In *Proc. of ACL*, pages 1127–1138.

Wolfgang Maier and Timm Lichte. 2016. Discontinuous parsing with continuous trees. In *Proc. of Workshop on Discontinuous Structures in NLP*, pages 47–57.

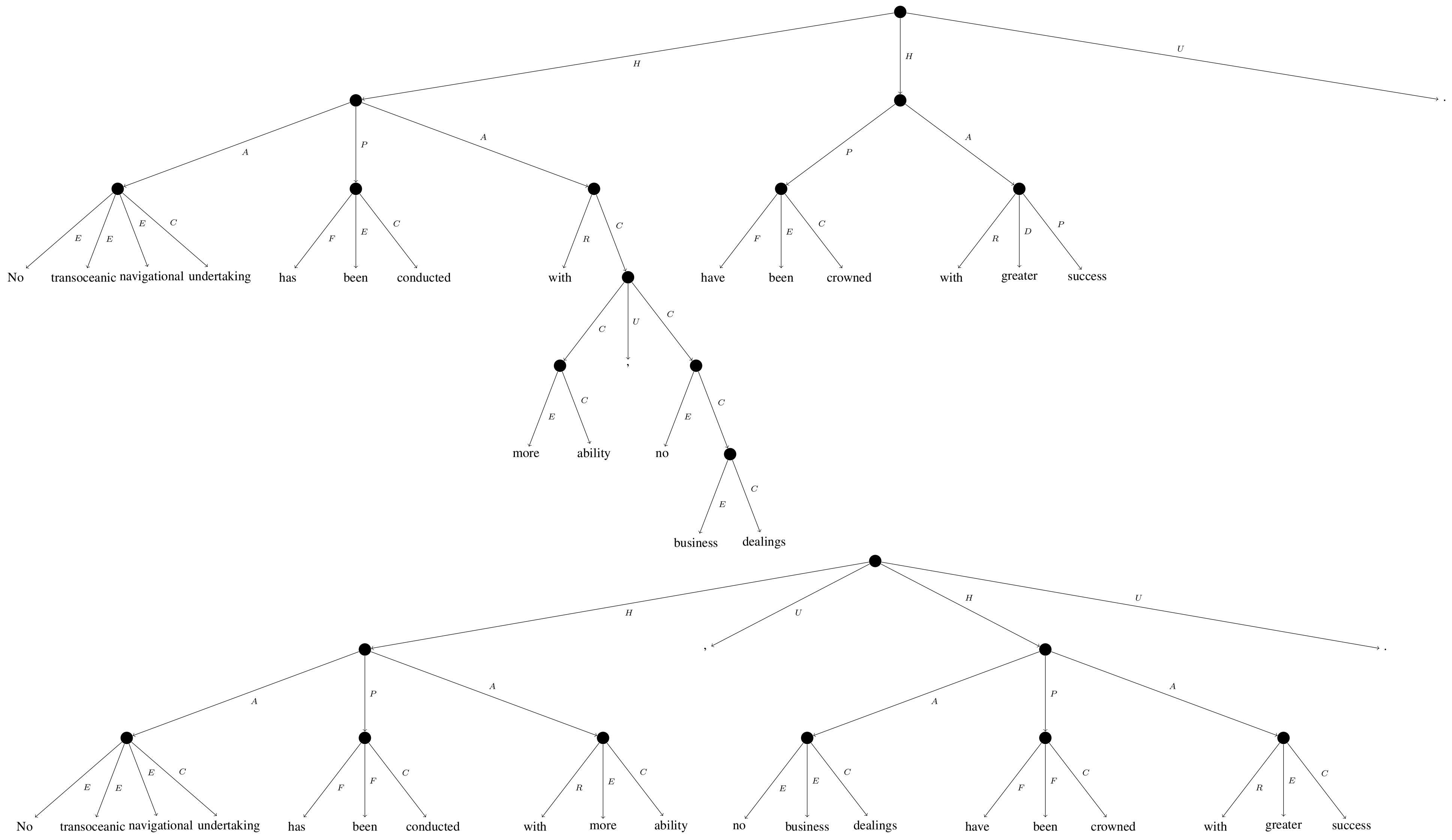

Figure 1: Output of single-task model on sentence 53001 from the English 20K test set (top), and of MTL model using all of AMR, DM and UD^{++} as auxiliaries on the same sentence (bottom).