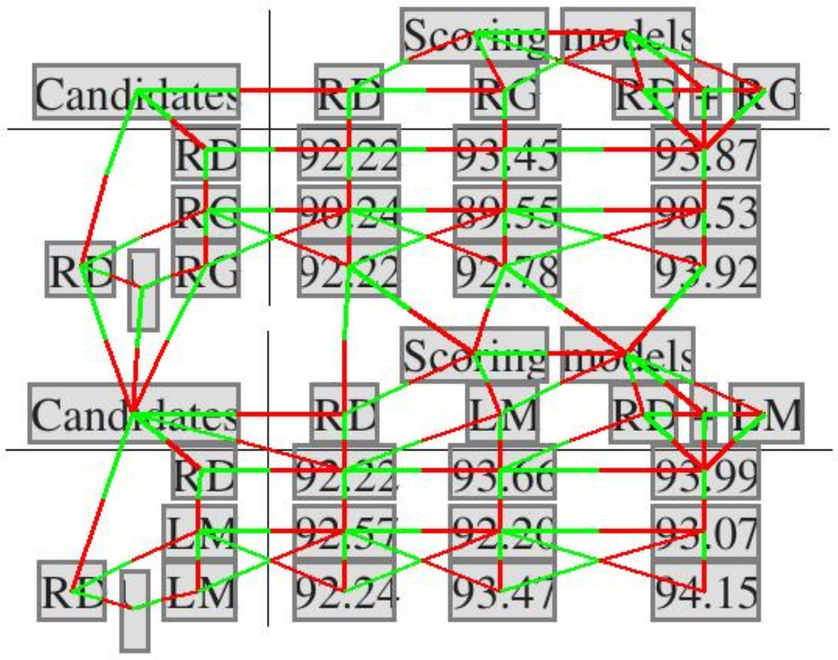

Table 2: Development F1 scores on section 22 of the PTB when using various models to produce candidates and to score them. ∪ denotes taking the union of candidates from each of two models; + denotes using a weighted average of the models' log-probabilities.

choose the top scoring candidate from these under A. We extend this by also searching directly in A to find high-scoring candidates for each sentence, and combining them with the candidate list proposed by B by taking the union, A ∪ B. We then choose the highest scoring candidate from this list under A. If A generally prefers parses outside of the candidate list from B, but these decrease evaluation performance (i.e., if B ∪ A → A is worse than B → A), this suggests a model combination effect is occurring: A makes errors which are hidden by having a limited candidate list from B.

This does seem to be the case for both generative models, as shown in Table 2, which presents F1 scores on the development set when varying the models used to produce the candidates and to score them. Each row is a different candidate set, where the third row in each table presents results for the augmented candidate sets; each column is a different scoring model, where the third column is the *score combination* setting described below. Going from RD → RG to the augmented candidate setting RD ∪ RG → RG decreases performance from 93.45 F1 to 92.78 F1 on the development set. This difference is statistically significant at the $p < 0.05$ level under a paired bootstrap test. We see a smaller, but still significant, effect in the case of LM: RD → LM achieves 93.66, compared to 93.47 for RD ∪ LM → LM.

We can also consider the performance of RG → RG and LM → LM (where we do not use candidates from RD at all, but return the highest-scoring parse from searching directly in one of the generative models) as an indicator of reranking effects: absolute performance is higher for LM (92.20 F1) than for RG (89.55). Taken together, these results suggest that model combination contributes to the success of both models, but to a larger extent for RG. A reranking effect may be a larger contributor to the success of LM, as this model achieves stronger performance on its own for the described search setting.

3.2 Score combination

If the cross-scoring setup exhibits an implicit model combination effect, where strong performance results from searching in one model and scoring with the other, we might expect substantial further improvements in performance by explicitly combining the scores of both models. To do so, we score each parse by taking a weighted sum of the log-probabilities assigned by both models (Hayashi et al., 2013), using an interpolation parameter which we tune to maximize F1 on the development set.

These results are given in columns RD + RG and RD + LM in Table 2. We find that combining the scores of both models improves on using the score of either model alone, regardless of the source of candidates. These improvements are statistically significant in all cases. Score combination also more than compensates for the decrease in performance we saw previously when adding in candidates from the generative model: RD ∪ RG → RD + RG improves upon both RD → RG and RD ∪ RG → RG, and the same effect holds for LM.

3.3 Strengthening model combination

Given the success of model combination between the base model and a single generative model, we also investigate the hypothesis that the generative models are complementary. The Model Combination block of Table 3 shows full results on the test set for these experiments, in the PTB column. The same trends we observed on the development data, on which the interpolation parameters were tuned, hold here: score combination improves results for all models (row 3 vs. row 2; row 6 vs. row 5), with candidate augmentation from the generative models giving a further increase (rows 4 and 7).[2] Combining candidates and scores from all three models (row 9), we obtain 93.94 F1.

[2]These increases, from adding score combination and candidate augmentation, are all significant with $p < 0.05$ in the PTB setting. In the +S data setting, all are significant except for the difference between row 5 and row 6.