AMRs Assemble!
Learning to Ensemble with Autoregressive Models for AMR Parsing

Abelardo Carlos Martínez Lorenzo[1,2*] **Pere-Lluís Huguet Cabot**[1,2*]
Roberto Navigli[2]
[1] Babelscape, Italy
[2] Sapienza NLP Group, Sapienza University of Rome
{martinez,huguetcabot}@babelscape.com
navigli@diag.uniroma1.it

Abstract

In this paper, we examine the current state-of-the-art in AMR parsing, which relies on ensemble strategies by merging multiple graph predictions. Our analysis reveals that the present models often violate AMR structural constraints. To address this issue, we develop a validation method, and show how ensemble models can exploit SMATCH metric weaknesses to obtain higher scores, but sometimes result in corrupted graphs. Additionally, we highlight the demanding need to compute the SMATCH score among all possible predictions. To overcome these challenges, we propose two novel ensemble strategies based on Transformer models, improving robustness to structural constraints, while also reducing the computational time. Our methods provide new insights for enhancing AMR parsers and metrics. Our code is available at github.com/babelscape/AMRs-Assemble.

1 Introduction

Semantic Parsing is the subfield of Natural Language Understanding (Navigli, 2018) that aims to encode the meaning of a sentence in a machine-interpretable structure. One of the formalisms that has gained more attention is the Abstract Meaning Representation (Banarescu et al., 2013, AMR), which embeds the semantics of a sentence in a directed acyclic graph. In AMR, concepts are represented by nodes, and semantic relations between concepts by edges (see Figure 1). AMR parsing has been applied to various areas of NLP, including Question Answering (Lim et al., 2020; Bonial et al., 2020; Kapanipathi et al., 2021), Text Summarization (Hardy and Vlachos, 2018; Liao et al., 2018), Information Extraction (Rao et al., 2017), and Machine Translation (Song et al., 2019), and has been extended to non-English languages (Anchiêta and Pardo, 2020; Blloshmi et al., 2020; Oral and Eryiğit, 2022; Navigli et al., 2022; Martínez Lorenzo et al., 2022).

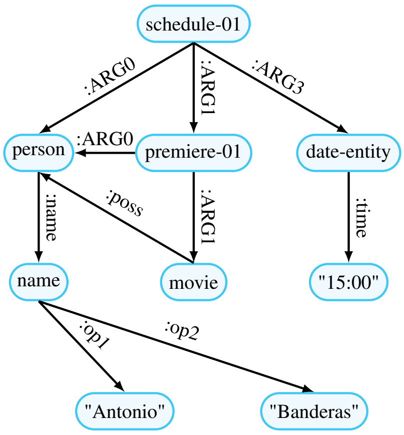


Figure 1: AMR graph of the sentence: *"Antonio Banderas scheduled the premiere of his movie at 3 pm".*

Current AMR parsing approaches are based on Transformer sequence-to-sequence (seq2seq) models (Bevilacqua et al., 2021, SPRING), which translate text into a linearized representation of the AMR graph. Recently, there have been some improvements through techniques such as pre-training on structural graph information (Bai et al., 2022), incorporating shallow semantic information (Chen et al., 2022), modifying ancestor information during decoding (Yu and Gildea, 2022), and adding a structural graph prediction task during training (Cheng et al., 2022). Nevertheless, in an attempt to push SMATCH (Cai and Knight, 2013) performance, there has been a recent trend towards ensemble models, which merge AMR graph predictions from multiple systems. Some examples include Graphene (Lam et al., 2021), a graph mining algorithm that searches for the largest common structure among the graph predictions, or the Maximum Bayes SMATCH Ensemble (Lee et al., 2022), which introduces a Bayesian ensemble approach in order to create high-quality silver data. However, notwithstanding their higher performance, ensem-

[*] Equal contributions.

Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 1595–1605
July 9-14, 2023 ©2023 Association for Computational Linguistics

ble models are potentially more vulnerable to producing corrupted AMR graphs. For instance, Opitz and Frank (2022) highlighted that better SMATCH scores do not always correlate with better parsing.

In this study, we conduct an investigation into the reasons why ensemble models improve their performance and in which cases they do so despite producing corrupted output. Our analysis reveals three significant drawbacks in these approaches: *i)* ensemble systems do not consider structural constraints in AMR, treating AMR graphs as regular sets of triplets, *ii)* they rely on SMATCH, which does not impose AMR constraints, exacerbating the problem of corrupted AMR graphs produced by ensemble methods that prioritize a higher score over adherence to structural constraints, as is the case with Graphene, and *iii)* they are computationally expensive. Our findings highlight the need for more robust evaluation metrics that hold to the structural constraints of AMR.

In this paper, we present two novel ensemble strategies that address the above limitations of current approaches. In the first strategy, we follow previous *merging* methods, showing how to train a seq2seq model to combine different predictions by taking into account both the original sentence and predictions from multiple models. In our second approach, we propose using *selection* as the ensembling strategy, where we select the best graph instead of merging. Specifically, we base our method on the perplexity score of the model. Additionally, we propose a graph algorithm that checks the structural constraints in AMR graphs. Through these contributions, we aim to provide more robust and efficient solutions for ensembling AMRs.

2 AMRs Assemble!

The task of AMR parsing can be framed as a seq2seq task, where the input $t = [x_1, x_2, ..., x_m]$ is a sequence of m tokens and the output $g = [g_1, g_2, ..., g_n]$ is a linearized graph with n tokens. To illustrate, the linearized representation of the AMR graph in Figure 1 is as follows:

```
(z0 / schedule-01
    :ARG0 (z1 / person
        :name (z2 / name
            :op1 "Antonio"
            :op2 "Banderas"))
    :ARG1 (z3 / premiere-01
        :ARG0 z1
        :ARG1 (z4 / movie
            :poss z1)
    :time (z5 / date-entity
        :time "15:00")))
```

The goal of seq2seq AMR parsing task is to learn a function that models the conditional probability:

$$p(g|x) = \prod_{t=1}^{n} p(e_t|e_{<t}, x), \tag{1}$$

where $e_{<t}$ are the tokens of the linearized graph g before step t.

In this work, we use LongT5 (Guo et al., 2022) as the seq2seq model, which is specialized for long sequences, making it feasible to provide sentences and linearized graphs as input.

2.1 Pre-training

To enhance the structure awareness of the language model in relation to AMR graphs and ensembling techniques, we extend the graph self-supervised pre-training method proposed by Bai et al. (2022, AMRBart). Formally, we denote a sentence as $t = [x_1, x_2, ..., x_m]$, a graph as $g = [g_1, g_2, ..., g_n]$, and a prediction by system s as $p_s = [p^s_1, p^s_2, ..., p^s_{l_s}]$. We follow AMRBart noise function with a dynamic masking rate and denote the noisy text and graph as $\hat{t}$ and $\hat{g}$, respectively. Moreover, let $\bar{t}$, $\bar{g}$, and $\bar{p}$ be the empty text, graph and prediction, respectively. As shown in Table 1, our pre-training procedure includes tasks presented by AMRBart, such as: i) empty text graph denoising ($\bar{t}\hat{g}2g$), ii) text augmented graph denoising ($t\hat{g}2g$), and iii) noisy text augmented graph denoising ($\hat{t}\hat{g}2g$). Additionally, we introduce: iv) empty text multiple graph denoising $\bar{t}\hat{g}_1...\hat{g}_k2g$, where the target graph is generated using different graphs' masked versions, and v) noisy text augmented multiple graph denoising $\hat{t}\hat{g}_1...\hat{g}_k2g$, where we also include the masked sentence.

2.2 Fine-tuning

Prediction Corpus To fine-tune ensemble systems we create a corpus of multiple predictions, starting from AMR 3.0 (LDC2020T02), which consists of 59,255 human-annotated sentence-graph pairs. We create five distinct train-test splits of this dataset in such a way that each test set is one fifth of the data and mutually exclusive. We train five separate models, based on Blloshmi et al. (2021), on the corresponding training sets and use each model to generate predictions for its respective test set. By combining all of the predicted test sets, we obtain a corpus of AMR predictions. However, to train an ensemble model, it is necessary to merge predictions from multiple models. Therefore, we

Phase	Task	Input	Output
Pre-training	$\bar{t}\hat{g}2g$	<s> $[mask]$ <g> $g_1, ...[mask]..., g_n$ </s>	<s> $g_1, g_2, ..., g_n$ </s>
	$t\hat{g}2g$	<s> $x_1, x_2, ..., x_m$ <g> $g_1, ...[mask]..., g_n$ </s>	<s> $g_1, g_2, ..., g_n$ </s>
	$\hat{t}\hat{g}2g$	<s> $x_1, ...[mask]..., x_m$ <g> $g_1, ...[mask]..., g_n$ </s>	<s> $g_1, g_2, ..., g_n$ </s>
	$\bar{t}\hat{g}_1...\hat{g}_k2g$	<s> $[mask]$ <g> $g_1, ...[mask]_1..., g_n$ <g> ... <g> $g_1, ...[mask]_k..., g_n$ </s>	<s> $g_1, g_2, ..., g_n$ </s>
	$\hat{t}\hat{g}_1...\hat{g}_k2g$	<s> $x_1, ...[mask]..., x_m$ <g> $g_1, ...[mask]_1.., g_n$ <g> ... <g> $g_1, ...[mask]_k..., g_n$ </s>	<s> $g_1, g_2, ..., g_n$ </s>
Fine-tun.	$t\overline{p}_{1...k}2g$	<s> $x_1, x_2, ..., x_m$ <g> $[mask]$ </s>	<s> $g_1, g_2, ..., g_n$ </s>
	$\bar{t}p_{1...k}2g$	<s> $[mask]$ <g> $p^1_1, p^1_2, ..., p^1_{l_1}$ <g> ... <g> $p^k_1, p^k_2, .., p^k_{l_k}$ </s>	<s> $g_1, g_2, ..., g_n$ </s>
	$tp_{1...k}2g$	<s> $x_1, x_2, ..., x_m$ <g> $p^1_1, p^1_2, ..., p^1_{l_1}$ <g> ... <g> $p^k_1, p^k_2, .., p^k_{l_k}$ </s>	<s> $g_1, g_2, ..., g_n$ </s>

Table 1: Pre-training and fine-tuning tasks. t denotes text, g denotes graph, p denotes prediction.

generate five distinct prediction corpora by repeating this process four additional times with different train-test split sets.

Strategy Having obtained a corpus comprising multiple AMR predictions, we design a set of tasks that fine-tune the model for ensembling. The first task is AMR parsing ($t\overline{p}_{1...k}2g$), i.e., an AMR graph g is generated by using only a sentence t as input. In the second task, ensemble AMR predictions ($\bar{t}p_{1...k}2g$), the model is provided with a random set of AMR predictions p without the corresponding sentence, so it is forced to use just graph information to ensemble. In the last task, ensemble AMR predictions using the sentence ($tp_{1...k}2g$), the model is provided with both a random set of AMR predictions p and the original sentence t. To ensure that the model is able to learn to merge a variety of predictions, we randomly modify the samples by changing the order and number of predictions in each epoch. As a result of this process, we obtain a model that is able to effectively integrate information from multiple sources to generate high-quality AMR graphs, without relying on the expensive SMATCH metric as has been the case for previous ensemblers.

2.3 Assemble! zero & avg

Nevertheless, using large autoregressive models to generate AMR graphs can be computationally expensive. Therefore, we propose an alternative approach that is more effective than previous merging strategies. Our method selects the best graph from a set of predictions. To achieve this, we introduce two novel scoring functions, in which we provide each predicted graph to the decoder of a model and extract their perplexity score, which can be done with a single forward pass. In the first method (Assemble!$_{zero}$), we leverage our trained ensemble model by providing the sentence and all the predictions in order to extract their perplexities and select the smallest one, i.e., we select prediction $p_{s'}$, where:

$$s' = \operatorname*{argmin}_{s \in \{1,\dots,l\}} perplexity(tp_{1...l}2p_s).$$

In the second method (Assemble!$_{avg}$), instead of using our ensembler, we use each model that generated the predictions to extract the perplexity for all the candidates. The final output is the graph $p_{s'}$ with the lowest average perplexity score, where:

$$s' = \operatorname*{argmin}_{s \in \{1,\dots,l\}} \frac{1}{l} \sum_{j \in \{1,\dots,l\}} perplexity_j(t2p_s).$$

3 Experiments

3.1 Setup

Dataset We evaluate our model using AMR 3.0. For pre-training, we use the same $200k$ silver data parsed by Bevilacqua et al. (2021, SPRING) from the Gigaword *(LDC2011T07)* corpus. For fine-tuning, we use the corpus described in Section 2.2.

Metric To evaluate our results, we employ the SMATCH metric, which quantifies the similarity between graphs by measuring the degree of overlap between their triplets, and SMATCH's breakdown metrics (see Appendix D). In addition, we validate our results using two novel AMR metrics: S^{2}MATCH (Opitz et al., 2020) and WWLK (Opitz et al., 2021), in its WWLK-k3e2n version introduced in Opitz et al. (2021).

Ensemble Baselines For our selection strategy, we use the system of Barzdins and Gosko (2016) as a baseline, which calculates the average SMATCH score for a given graph in comparison to all the other candidates and selects the one with the highest score.

Our baseline for merging is Graphene (Lam et al., 2021), an algorithm that identifies the graph with the most nodes and edges in common among

	Model	Time (s)	Corrupt.	SMATCH	S2MATCH	WWLK	Unlab.	NoWSD	Conc.	NER	Neg.	Wiki	Reent.	SRL
Predictions	SPRING$_1$	—	54	83.1	84.3	84.9	86.2	83.6	89.3	87.7	70.9	81.5	72.9	81.8
	SPRING$_2$	—	52	82.7	83.9	82.1	85.9	83.2	89.0	87.5	72.6	80.2	73.0	81.4
	SPRING$_3$	—	73	83.0	84.3	85.1	86.3	83.5	89.3	87.6	72.6	81.8	73.1	81.7
	SPRING$_4$	—	33	82.8	84.0	84.1	86.0	83.3	88.9	87.3	71.7	81.5	72.8	81.4
	SPRING$_5$	—	104	82.6	83.9	84,5	85.8	83.2	89.2	87.3	73.0	81.6	73.0	81.4
	Best$_{graph}$	—	51	86.5	87.5	88.0	89.0	86.9	91.7	89.9	76.5	83.8	77.7	85.4
Mergers	Graphene$_{base}$	810	374	83.6	84.8	84.9	86.6	84.1	89.8	88.0	73.5	81.2	72.3	82.4
	Graphene$_{SMATCH}$	11,884	260	**83.8**	**85.0**	85.0	86.9	**84.4**	**89.9**	88.1	**73.8**	81.3	73.7	**82.6**
	Assemble!	**431**	**6**	**83.8**	**85.0**	**85.2**	**87.0**	84.3	89.7	**88.3**	72.9	**81.7**	**74.2**	82.3
Selectors	SMATCH$_{avg}$	493	51	83.7	85.0	85.3	86.8	84.2	89.7	88.1	73.3	82.0	73.9	82.4
	Assemble!$_{zero}$	**256**	**13**	83.9	85.1	**85.4**	87.1	84.4	**89.9**	**88.3**	**74.0**	**82.2**	74.3	82.5
	Assemble!$_{avg}$	635	22	**84.1**	**85.3**	84.4	**87.2**	**84.6**	**89.9**	**88.3**	73.3	**82.2**	**74.6**	**82.8**

Table 2: Results in AMR 3.0 test set. Bold indicates best. Columns: Model, computational time, corrupted graphs, SMATCH, S^{2}MATCH, WWLK and SMATCH breakdown. Row Blocks: Predictions, Best predicted and models.

different graphs. Specifically, given a pivot graph g_i (where $i = 1, 2, ..., k$), Graphene collects votes from the other graphs for every existing vertex and existing/non-existing edges to correct g_i. We use two variants of Graphene, i) Graphene$_{base}$, where every input graph is chosen as a pivot graph once, and the best among the modified pivot graphs is chosen as the final prediction based on average support; and ii) Graphene$_{smatch}$, which is similar to Graphene$_{base}$ but chooses the best modified pivot graph based on average SMATCH score, similar to Barzdins and Gosko (2016).

We do not compare our approach using Maximum Bayes SMATCH Ensemble (Lee et al., 2022), as it is a technique for producing high-quality silver data by combining SMATCH-based ensembling techniques with ensemble distillation, and its code and data are not publicly available.

Our Models We simulate an ensemble of five models obtained by training SPRING on five different seeds, and apply these models to the test split of AMR 3.0 using each of them. Assemble! and Assemble!$_{zero}$ rely on LongT5 (Guo et al., 2022) and are trained as explained in Section 2.

3.2 Results

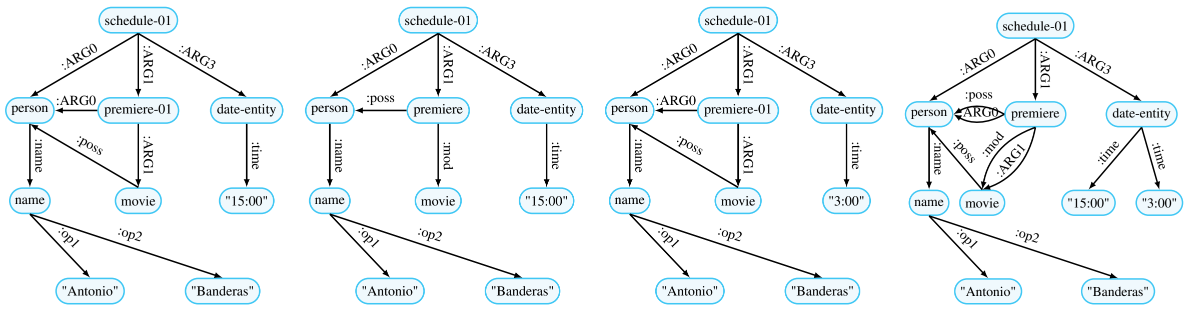
We present our results in Table 2. The *Predictions* block shows the performance of each individual system used for ensembling, which have an average SMATCH score of 82.8. The *Best*$_{graphs}$ row portrays the upper bound of the selection strategy, where the SMATCH score is calculated with an oracle that selects the graph with the highest SMATCH. This score is 3.4 points higher than the best predictions. The *Mergers* block presents the results of the ensembling strategies that combine predictions, where we observe that our model performs comparably to Graphene$_{smatch}$ but is 10 times faster. Furthermore, the *Selector* block presents the results of the three different selection strategies, where the best graph is chosen out of a set of predictions. Our strategy outperforms SMATCH$_{avg}$ by 0.4 points while having a similar computation time. These results demonstrate the effectiveness of our proposed ensembling approaches and suggest that they may be an alternative to traditional merging methods.

3.3 Analysis

While our model is able to effectively ensemble graphs or select the most accurate one from a set of predictions in an efficient and competitive manner, it is important to note that a higher SMATCH score does not always equate to the best graph if the graph has structural issues. This is because the SMATCH metric simply views the graph as a set of triplets. For example, the AMR graph illustrated in Figure 2(a) is treated as the following triplets:

```
(empty, :root, z0) ^
(z0, :instance, schedule-01) ^
(z0, :ARG0, z1) ^
(z1, :instance, person) ^
(z1, :name, z2) ^
(z2, :instance, name) ^
(z2, :op1, "Antonio") ^
(z2, :op2, "Banderas") ^
(z0, :ARG1, z3) ^
(z3, :instance, premiere-01) ^
(z3, :ARG0, z1) ^
(z3, :ARG1, z4) ^
(z4, :instance, movie) ^
(z4, :poss, z1) ^
(z0, :ARG3, z5) ^
(z5, :instance, date-entity) ^
(z5, :time, "15:00")
```

SMATCH calculates the degree of overlapping between two sets of triplets, but it does not consider the implicit AMR constraints. To address this problem, we develop an algorithm that checks some AMR violations in graphs: *i)* non-predicate nodes with `:ARG` relations, *ii)* predicate nodes with `:op` or `:snt` relations, *iii)* compositional issues in entity



(a) Gold AMR

(b) Pred 1. SMATCH 80,0.

(c) Pred 2. SMATCH 88,9.

(d) Graphene. SMATCH 85,0.

Figure 2: AMR representations of the sentence: *"Antonio Banderas scheduled the premiere of his movie at 3 pm"*.

structures, and *iv)* compositional issues in connector structures. The *Corrupt.* column in Table 2 shows the number of graphs with structural problems out of 1898 graphs. This highlights the limitation of previous ensemblers, such as Graphene, which do not consider these structural constraints.

Ensembling As demonstrated in the *Corrupt.* column of Table 2, our ensemble method has a significantly lower number of graphs with structural issues (0.3%) as compared to Graphene$_{base}$ and Graphene$_{smatch}$ (13.7-19.7%). This is because previous ensemble models are only focused on achieving a higher SMATCH metric, interpreting the graphs just as a set of triplets. This leads to ensembled graphs with violations of AMR guidelines and semantic inconsistencies. Figure 2(d) shows the Graphene$_{smatch}$ generated graph, and Figures 2(b) and 2(c) the two predictions used for ensembling. The Graphene$_{smatch}$ graph presents multiple AMR violations that are not in its predictions, e.g., the *premiere* node is connected to the *movie* node with two different relations because Graphene cannot decide which is the correct edge (both relations have the same probability), and one of the relations is an argument relation (i.e., :ARG), which cannot be used with non-predicate nodes since their meaning is encoded in PropBank frames.

SMATCH Graphene results in Table 2 are competitive despite having a higher percentage of structural issues in the ensembled graphs. This discrepancy can be attributed to the inherent properties of the SMATCH metric, which penalizes missing triplets more than wrong triplets. For example, the ensembled graph of Figure 2(d) obtains a higher SMATCH score than the prediction of Figure 2(b), since, in case of doubt, selecting both triplets (relations $ARG1$ and mod) from node *premiere* to node *movie* results in a higher score than selecting only the wrong triplet. This illustrates how current ensemble models exploit SMATCH weaknesses to attain higher scores. In contrast, our approaches provide competitive results while also being more robust to AMR constraints.

Furthermore, as highlighted in Opitz and Frank (2022), the current scores of AMR parsers and ensemblers (around 0.83 and 0.84, respectively) are higher than the average annotator vs. consensus inter-annotator agreement reported in Banarescu et al. (2013) (0.83 and 0.79 in newswire and web text, respectively). Additionally, WWLK results in Table 2 show how SPRING$_3$ predictions achieve comparable results to all ensemble models. Therefore, given the issues discussed above, the suitability of SMATCH for evaluating the model's performance beyond 0.83 has to be called into question.

4 Conclusion

In this paper, we leveraged self-supervised pretraining and a denoising autoencoder architecture to achieve strong results in merging AMR graph predictions. We also introduced two novel approaches for ensembling that select the best prediction from a set of candidates using simple and efficient perplexity score functions. These results suggest that the selection strategy is a promising alternative for ensembling, since it achieves competitive performance while being less expensive.

Furthermore, we developed an algorithm that checks the structural AMR constraints in parsing outputs. This allowed us to perform an analysis that revealed how previous ensemble models produce higher score graphs but exploit SMATCH weaknesses that lead to increased structural issues. Overall, our findings highlight the need for more robust evaluation metrics and ensemble models that are designed to adhere to the structural constraints.

5 Limitations

Our proposed ensemble approach for training the Transformer architecture has demonstrated promising results for the task of AMR ensembling. However, there are limitations that warrant further investigation in future research.

Our first limitation is the lack of generalization, as the approach was only evaluated on AMR parsing. Therefore, the application of an autoregressive ensembling model has not yet been tested on other Natural Language Processing tasks.

Moreover, in order to properly compare each ensemble system under the same conditions, we base all our experiments using the same underlying architecture, i.e. SPRING. There needs to be an exploration of these approaches using more recent, better performing parsers. However, this will require access to such systems.

Furthermore, the computational cost is also a limitation, as even though our proposed merger method, Assemble!, is more efficient than previous ensemblers, it is still computationally expensive, and particularly when we have to ensemble long graphs from multiple predictions. Moreover, as our Assemble! model is based on LongT5, it might be challenged when working with large datasets or when running experiments on resource-constrained systems. Therefore, we encourage the use of ensembling strategies focused on selecting the best graphs instead of merging.

Lastly, as our ensemble approach is based on Transformer, results can be difficult to interpret, as it can be challenging to understand how the generated graph has been ensembled by different predictions, leading to a lack of interpretability.

In summary, the proposed ensemble approach for training the Transformer architecture has shown promising results for the task of AMR ensembling and has the potential to be applied to other tasks, however, further research is necessary to address its limitations and improve performance.

6 Ethics Statement

Regarding the ethical and social implications of our approach for AMR ensembling, we do not believe it could have a negative impact. However, since ethical considerations are an important aspect of any research and development project, we will discuss a few ethical considerations here.

First, one potential concern is the use of Transformer-based models, which have been shown to perpetuate societal biases present in the data used for training. Our approach relies on the use of these models, and it is crucial to ensure that the data used for training is diverse and unbiased.

Second, it is important to consider the potential impact of the proposed ensemble strategies on marginalized communities. It is possible that these strategies may inadvertently perpetuate or amplify existing biases in the data used to train and test these systems. Therefore, it is important to ensure that the proposed ensemble strategies are tested on a diverse set of data and that any biases are identified and addressed.

In conclusion, the proposed ensemble strategies in this paper can potentially have positive impact on the field of AMR parsing, however, it is important to consider the ethical implications of this research and take steps to mitigate any potential negative consequences.

Acknowledgments

The authors gratefully acknowledge the support of the European Union's Horizon 2020 research project *Knowledge Graphs at Scale* (KnowGraphs) under the Marie Marie Skłodowska-Curie grant agreement No 860801.

The last author gratefully acknowledges the support of the PNRR MUR project PE0000013-FAIR. The authors sincerely thank Lorenzo Proietti and Stefano Perrella for their contribution to this project.

References

Rafael Anchiêta and Thiago Pardo. 2020. Semantically inspired AMR alignment for the Portuguese language. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1595–1600, Online. Association for Computational Linguistics.

Xuefeng Bai, Yulong Chen, and Yue Zhang. 2022. Graph pre-training for AMR parsing and generation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6001–6015, Dublin, Ireland. Association for Computational Linguistics.

Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. 2013. Abstract Meaning Representation for sembanking. In *Proceedings of the 7th Linguistic*

Annotation Workshop and Interoperability with Discourse, pages 178–186, Sofia, Bulgaria. Association for Computational Linguistics.

Guntis Barzdins and Didzis Gosko. 2016. RIGA at SemEval-2016 task 8: Impact of Smatch extensions and character-level neural translation on AMR parsing accuracy. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 1143–1147, San Diego, California. Association for Computational Linguistics.

Michele Bevilacqua, Rexhina Blloshmi, and Roberto Navigli. 2021. One SPRING to Rule Them Both: Symmetric AMR semantic Parsing and Generation without a Complex Pipeline. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(14):12564–12573.

Rexhina Blloshmi, Michele Bevilacqua, Edoardo Fabiano, Valentina Caruso, and Roberto Navigli. 2021. SPRING Goes Online: End-to-End AMR Parsing and Generation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 134–142, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Rexhina Blloshmi, Rocco Tripodi, and Roberto Navigli. 2020. XL-AMR: enabling cross-lingual AMR parsing with transfer learning techniques. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 2487–2500. Association for Computational Linguistics.

Claire Bonial, Stephanie M. Lukin, David Doughty, Steven Hill, and Clare Voss. 2020. InfoForager: Leveraging semantic search with AMR for COVID-19 research. In *Proceedings of the Second International Workshop on Designing Meaning Representations*, pages 67–77, Barcelona Spain (online). Association for Computational Linguistics.

Shu Cai and Kevin Knight. 2013. Smatch: an evaluation metric for semantic feature structures. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 748–752, Sofia, Bulgaria. Association for Computational Linguistics.

Liang Chen, Peiyi Wang, Runxin Xu, Tianyu Liu, Zhifang Sui, and Baobao Chang. 2022. ATP: AMRize then parse! enhancing AMR parsing with PseudoAMRs. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 2482–2496, Seattle, United States. Association for Computational Linguistics.

Ziming Cheng, Zuchao Li, and Hai Zhao. 2022. BiBL: AMR parsing and generation with bidirectional Bayesian learning. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 5461–5475, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Marco Damonte, Shay B. Cohen, and Giorgio Satta. 2017. An incremental parser for Abstract Meaning Representation. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 536–546, Valencia, Spain. Association for Computational Linguistics.

Mandy Guo, Joshua Ainslie, David Uthus, Santiago Ontanon, Jianmo Ni, Yun-Hsuan Sung, and Yinfei Yang. 2022. LongT5: Efficient text-to-text transformer for long sequences. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 724–736, Seattle, United States. Association for Computational Linguistics.

Hardy Hardy and Andreas Vlachos. 2018. Guided neural language generation for abstractive summarization using Abstract Meaning Representation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 768–773, Brussels, Belgium. Association for Computational Linguistics.

Pavan Kapanipathi, Ibrahim Abdelaziz, Srinivas Ravishankar, Salim Roukos, Alexander Gray, Ramón Fernandez Astudillo, Maria Chang, Cristina Cornelio, Saswati Dana, Achille Fokoue, Dinesh Garg, Alfio Gliozzo, Sairam Gurajada, Hima Karanam, Naweed Khan, Dinesh Khandelwal, Young-Suk Lee, Yunyao Li, Francois Luus, Ndivhuwo Makondo, Nandana Mihindukulasooriya, Tahira Naseem, Sumit Neelam, Lucian Popa, Revanth Gangi Reddy, Ryan Riegel, Gaetano Rossiello, Udit Sharma, G P Shrivatsa Bhargav, and Mo Yu. 2021. Leveraging Abstract Meaning Representation for knowledge base question answering. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3884–3894, Online. Association for Computational Linguistics.

Hoang Thanh Lam, Gabriele Picco, Yufang Hou, Young-Suk Lee, Lam M. Nguyen, Dzung T. Phan, Vanessa López, and Ramon Fernandez Astudillo. 2021. Ensembling Graph Predictions for AMR Parsing.

Young-Suk Lee, Ramón Astudillo, Hoang Thanh Lam, Tahira Naseem, Radu Florian, and Salim Roukos. 2022. Maximum Bayes Smatch ensemble distillation for AMR parsing. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5379–5392, Seattle, United States. Association for Computational Linguistics.

Kexin Liao, Logan Lebanoff, and Fei Liu. 2018. Abstract Meaning Representation for multi-document summarization. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1178–1190, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Jungwoo Lim, Dongsuk Oh, Yoonna Jang, Kisu Yang, and Heuiseok Lim. 2020. I know what you asked:

Graph path learning using AMR for commonsense reasoning. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 2459–2471, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Abelardo Carlos Martínez Lorenzo, Marco Maru, and Roberto Navigli. 2022. Fully-Semantic Parsing and Generation: the BabelNet Meaning Representation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1727–1741, Dublin, Ireland. Association for Computational Linguistics.

Roberto Navigli. 2018. Natural language understanding: Instructions for (present and future) use. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden*, pages 5697–5702. ijcai.org.

Roberto Navigli, Rexhina Blloshmi, and Abelardo Carlos Martinez Lorenzo. 2022. BabelNet Meaning Representation: A Fully Semantic Formalism to Overcome Language Barriers. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36.

Juri Opitz, Angel Daza, and Anette Frank. 2021. Weisfeiler-leman in the bamboo: Novel AMR graph metrics and a benchmark for AMR graph similarity. *Transactions of the Association for Computational Linguistics*, 9:1425–1441.

Juri Opitz and Anette Frank. 2022. Better Smatch = better parser? AMR evaluation is not so simple anymore. In *Proceedings of the 3rd Workshop on Evaluation and Comparison of NLP Systems*, pages 32–43, Online. Association for Computational Linguistics.

Juri Opitz, Letitia Parcalabescu, and Anette Frank. 2020. AMR similarity metrics from principles. *Transactions of the Association for Computational Linguistics*, 8:522–538.

K. Elif Oral and Gülşen Eryiğit. 2022. AMR alignment for morphologically-rich and pro-drop languages. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop*, pages 143–152, Dublin, Ireland. Association for Computational Linguistics.

Sudha Rao, Daniel Marcu, Kevin Knight, and Hal Daumé III. 2017. Biomedical event extraction using Abstract Meaning Representation. In *BioNLP 2017*, pages 126–135, Vancouver, Canada,. Association for Computational Linguistics.

Linfeng Song, Daniel Gildea, Yue Zhang, Zhiguo Wang, and Jinsong Su. 2019. Semantic neural machine translation using AMR. *Transactions of the Association for Computational Linguistics*, 7:19–31.

Ledell Wu, Fabio Petroni, Martin Josifoski, Sebastian Riedel, and Luke Zettlemoyer. 2020. Scalable zero-shot entity linking with dense entity retrieval. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6397–6407, Online. Association for Computational Linguistics.

Chen Yu and Daniel Gildea. 2022. Sequence-to-sequence AMR parsing with ancestor information. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 571–577, Dublin, Ireland. Association for Computational Linguistics.

A Model Hyper-Parameters

Table 3 lists hyperparameters and search space for the experiments with SPRING models and our Assemble!. The masking probabilities of the pre-training task are: i) $\bar{t}\hat{g}2g$ – 0.35%, ii) $t\hat{g}2g$ – 0.35%, iii) $\hat{t}\hat{g}2g$ – from 0.15% to 0.85% incrementing by epoch, iv) $\bar{t}\hat{g}_1...\hat{g}_k2g$ – 0.55%, and v) $\hat{t}\hat{g}_1...\hat{g}_k2g$ – 0.55%.

Group	Parameter	Values
Pre-training	Optimizer	Adafactor
	Batch size	1
	Dropout	0.2
	Attent. dropout	0.0
	Grad. accum.	32
	Weight decay	0.01
	LR	0.0001
	LR sched.	Inverse sqrt
	Beamsize	5
Fine-tuning	Optimizer	Adafactor
	Batch size	1.0
	Dropout	0.1
	Attent. dropout	0.0
	Grad. accum.	32.0
	Weight decay	0.01
	LR	0.00001
	LR	Constant
	Beamsize	5

Table 3: Final hyperparameters and search space for the experiments.

B Hardware and size of the model

We performed experiments on a single NVIDIA 3090 GPU with 64GB of RAM and Intel® Core™ i9-10900KF CPU. The total number of trainable parameters of SKD is 434,883,596. The pre-training phase on the silver data requires 168 hours, whereas fine-tuning requires 216 hours.

C BLINK

All systems from Table 2 use BLINK (Wu et al., 2020) for wikification. For this purpose, we used the $blinkify.py$ script from the SPRING repository.

D Metric

To evaluate the predictions, we use the SMATCH metric and the extra scores of Damonte et al. (2017): *i)* Unlabel, compute on the predicted graphs after removing all edge labels, *ii)* No WSD, compute while ignoring Propbank senses (e.g., duck-01 vs duck-02), *iii)* Wikification, F-score on the wikification (:wiki roles), *iv)* NER, F-score on the named entity recognition (:name roles), *v)* Negations, F-score on the negation detection (:polarity roles), *vi)* Concepts, F-score on the concept identification task, *vii)* Reentrancy, computed on reentrant edges only, *viii)* Semantic Role Labeling (SRL), computed on :ARG-i roles only.

E Data

The AMR 3.0 data used in this paper is licensed under the *LDC User Agreement for Non-Members* for LDC subscribers, which can be found here.

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
Section 5

☑ A2. Did you discuss any potential risks of your work?
Section 6

☑ A3. Do the abstract and introduction summarize the paper's main claims?
In the abstract and in the introduction section.

☑ A4. Have you used AI writing assistants when working on this paper?
Grammarly, we use to check the use of English of our paper

B ☑ Did you use or create scientific artifacts?

Section 2 and 3

☑ B1. Did you cite the creators of artifacts you used?
Yes, Section 1 and 3

☑ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
Appendix

☑ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
The licenses are self-explanatory and discussed in the Appendix.

☒ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
The dataset has been widely used, and was already scrutinised for personal information before our use.

□ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
Not applicable. Left blank.

☑ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
Section 2

C ☑ Did you run computational experiments?

Section 3

☑ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
Appendix

The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.

☑ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
Appendix

☑ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
Section 3

☑ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
We compare with previous approaches and use their implementation (Section 3)

D ☒ Did you use human annotators (e.g., crowdworkers) or research with human participants?
Left blank.

□ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
No response.

□ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
No response.

□ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
No response.

□ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
No response.

□ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
No response.