TrAinMR: an Annotator Training Website for Abstract Meaning Representation

Mina Yang, Shira Wein
Amherst College
Amherst, MA, United States
{miyang27, swein}@amherst.edu

Abstract

Abstract Meaning Representation (AMR) is a graph-based semantic representation which captures the core elements of meaning of a text. AMR has been incorporated into a variety of downstream tasks, which rely heavily on the availability of gold-annotated AMR corpora. While the annotation process is fairly lightweight, annotator training is still required even for linguists due to the extensive nature of the annotation guidelines and comprehensive set of roles. Therefore, all corpus development projects for AMR (and extensions of AMR) require the dataset curators to first train annotators. In this paper, we develop an online AMR annotation training system called TrAinMR in order to ease this training process and thus motivate the development of additional AMR corpora. The two main components of TrAinMR are (1) a written tutorial covering the basics of AMR annotation, and (2) an interactive practice module with corrective feedback. To measure the effectiveness of this tool, we conduct two pilot studies with five human annotators each. We find that the majority of annotators state their understanding of AMR improved as a result of TrAinMR, and some annotators show a positive trend in SMATCH scores after completing the practice module.

Keywords: semantics, annotation, abstract meaning representation

1. Introduction

The graph-based semantic representation Abstract Meaning Representation (AMR; Banarescu et al., 2013) captures the relationship between concepts in a sentence of the form "who did what to whom" (see Figure 1). AMR has been incorporated into a variety of downstream applications in areas such as machine translation (Wein and Schneider, 2024; Song et al., 2019) and summarization (Liao et al., 2018), leveraging prior efforts towards gold annotation of AMR.

While datasets such as AMR3.0 (Knight et al., 2020) are widely used, creating AMR corpora requires the work of human annotators, and for datasets containing over thousands of AMRs, the annotation process becomes costly in time and work. This is in large part due to the extensive nature of the AMR guidelines. While AMR annotation is fairly lightweight, in order to produce accurate annotations, AMR annotators must be trained to identify relationships in the text and produce the correct AMR relations. Furthermore, prior work investigating the automation of AMR production finds that relying on GPT models to automate the curation of AMR datasets is not feasible due to the frequency of mistakes in AMR annotation (Ettinger et al., 2023). Therefore, although AMR annotation by hand can be laborious, our goal is to make the training process faster by providing a ready-to-use training schema for any AMR project developers. TrAinMR is designed to lessen the barrier of entry to AMR annotation, enabling novice non-linguists to begin to contribute to annotation projects at scale. Our tool may also be helpful as a first step for crowdworkers, if the project developer selects to crowdsource AMRs and incorporate expert intervention for more complicated instances (Martin et al., 2020).[1]

Sentence: The cat has orange stripes.

AMR Graph:

```
(h / have-03
    :ARG0 (c / cat)
    :ARG1 (s / stripe)
        :mod (o / orange))
```

Figure 1: The AMR graph for the simple sentence "The cat has orange stripes." For the root `(h / have-03)`, `have` is the concept, `-03` is the sense number as identified via PropBank, and `h` is the variable name which enables coreferential relations (i.e., for concepts which are referred back to later in a graph).

To assist with the challenging process of training new annotators, we present TrAinMR (Training in Abstract Meaning Representation), a web-based tool that consists of two main components: a Tutorial module which introduces the fundamental guidelines of AMR annotation, and an interactive Annotation Practice module. The Annotation Practice module asks users to annotate 25 sentences, and provides immediate feedback by comparing the user's annotations to a gold reference annota-

[1]The AMR training website is available at https://acnlplab.github.io/amr-training-site/

Proceedings of The Seventh International Workshop on Designing Meaning Representations (DMR 2026) @ LREC 2026, pages 44–53
11 May 2026.

tion.

In order to analyze the tool's effectiveness, we conduct two pilot studies, in which we measure TrAinMR's impact on performance. The study involves 10 novice human annotators who first read the Tutorial, then annotate 25 sentences. We analyze their performance using SMATCH scores (Cai and Knight, 2013), qualitative analysis on individual annotations, and collect post-study feedback from annotators on their experience.

2. Related Work

AMR is a graph-structured semantic representation designed to generalize away from surface-level grammatical details in order to capture the core meaning of a text (Banarescu et al., 2013). AMR concepts are reflected as nodes in the directed, rooted graph, and edges represent the relationship between those concepts; edges are marked with relations including numbered arguments from PropBank (e.g., `:ARGn`) or named relations (e.g., `:location`). Numerous extensions of the AMR annotation schema have been introduced, such as the Uniform Meaning Representation (Van Gysel et al., 2021), BabelNet Meaning Representation (Martínez Lorenzo et al., 2022), Spatial AMR (Bonn et al., 2020), and Gesture-AMR (Brutti et al., 2022)– all of which also require training in AMR before being able to produce the specialized annotations.

Key AMR materials include the AMR guidelines (Banarescu et al., 2019) and an AMR Tutorial presentation from NAACL 2015 that discusses foundational AMR concepts through practice examples (Schneider et al., 2015). Existing annotation interfaces include the (now defunct) AMR Editor (Hermjakob, 2013), CAMRA (Cai et al., 2023), LiDARR (Cai et al., 2025), Anafora (Chen and Styler, 2013), and X-AMR (Ahmed et al., 2024). These tools aid AMR annotation on the document- and corpus-level in order to improve the quality of the AMR, but are all designed to serve as annotation interfaces, not to train annotators. To the best of our knowledge, TrAinMR is the first tool designed for annotation training, serving as a centralized on-boarding for new annotators. By standardizing the initial training process, TrAinMR reduces the burden on new annotation efforts by allowing annotators to complete the TrAinMR system first.

In order to evaluate annotator performance, we use SMATCH, which is a standard approach for comparing two AMR graphs for semantic similarity (Cai and Knight, 2013). SMATCH calculates semantic similarity by aligning the concepts of the two graphs and counting their matching triples. Then, it calculates the precision, recall, and F1-score. The final score is a single value ranging from 0 (no match) to 1 (perfect match). Thus, we are able to compare the user's submission against a gold reference, expert-annotated AMR graph. SMATCH score can also be used to measure inter-annotator agreement in order to validate annotator training and consistency of a dataset. SMATCH scores for inter-annotator agreement of released AMR datasets range from 0.71 to 0.89 SMATCH (Wein, 2025).

3. System Overview

TrAinMR consists of two main components: a Tutorial module and an Annotation Practice module.

The Tutorial page contains written guidance on how to annotate an AMR graph, including instructions on PENMAN notation (Matthiessen and Bateman, 1991), PropBank framesets (Palmer et al., 2005), and common roles and relations.

After reading the Tutorial, users are directed to then practice writing their annotations in the Annotation Practice page. On the Annotation Practice page, we instruct the users to submit their annotations for 25 curated sentences, and detail that the user can view the gold AMR and explanations for the AMR after submitting an annotation. External resources, such as the PropBank repository (Palmer et al., 2005) and the AMR guidelines (Banarescu et al., 2019), are referenced on the Annotation Practice and Tutorial pages, and users are encouraged during both stages to supplement their understanding. By practicing after learning about the AMR guidelines, users are able to apply and reinforce their learning through active recall. This also enables users to quickly identify and remedy misunderstandings from the Tutorial as they receive corrective feedback on their AMR graph submissions.

3.1. Tutorial

The content of the Tutorial page largely follows the list of topics in the AMR paper Banarescu et al. (2013) and AMR Tutorial Slides (Schneider et al., 2015). These topics are: AMR, PENMAN notation, PropBank frames, nominalization, AMR roles and relations (including frame arguments and common relations), reification (converting a relation into a concept), and annotation of negation, modals, and questions. Each topic in the Tutorial includes a brief explanation, as well as example sentences and their corresponding AMR graphs which demonstrate those topics (see Figure 2 for an example). The example sentences in the beginning of the Tutorial are short sentences designed to exemplify a specific topic, such as the application of `:mod`. There are supplemental examples at the bottom of the Tutorial that feature longer AMRs with a combination of previous topics, including inverse relations, modification, and quantification.

Figure 2: An example of the "Listing Entities" section from the Tutorial page is shown, including the AMR for the sentence "Funk and soul."

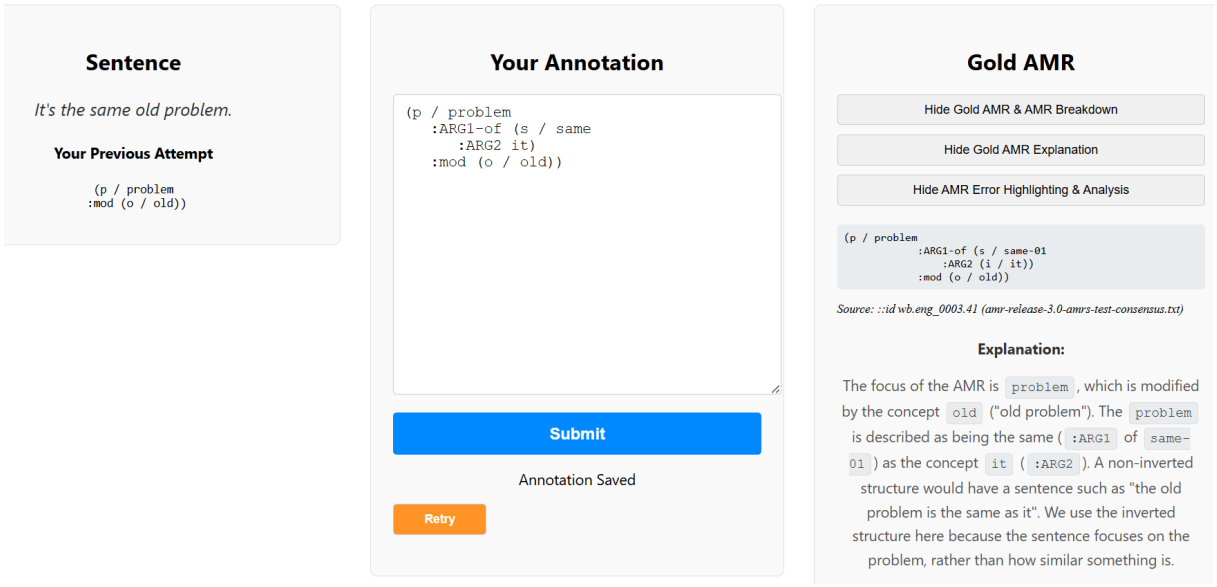

Figure 3: The Annotation Practice module displays feedback when prompted by the user. After selecting to show the Gold AMR, the AMR is shown as well as an explanation of the contents of the graph.

3.2. Annotation Practice

After having engaged with the Tutorial, the user is then able to try AMR annotation on the Annotation Practice page. We store the user's submissions in a Google Sheet and display the highlighted differences between the submission and the gold AMR. We also provide an explanation of the various components of the gold AMRs, including the root concept (the focus of the sentence), other concepts, and the roles and relations that connect them.

The gold AMR is hidden from view until an input is submitted. Once users submit an annotation, their attempt is shown below the sentence and they are able to resubmit if desired. To help users visually track their progress, their most recent submission for each sentence is saved and displayed.

Additionally, after submitting, users have the option to view the gold AMR and its line-by-line breakdown, the gold AMR explanation, and the AMR analysis separately.

3.2.1. User Interface

The Annotation Practice page consists of 25 practice annotations for sentences taken from the AMR3.0 dataset (Knight et al., 2020). We choose sentences manually to reflect a range of complexity and the topics mentioned on the Tutorial page. We define complexity using sentence length and the presence of combinations of topics, such as modals and inverse roles. For example, we consider a short sentence containing a single subject and verb to be simpler than a long sentence with multiple adjectives, subjects, verbs, and objects, which we consider to be more complicated to annotate.

The user can choose to show the sentences either in order of increasing complexity or in randomized order, which is the default.

Our Annotation Practice page consists of primarily three panels: the leftmost containing the sentence and an option for the user to view any previous submissions for that sentence, the middle

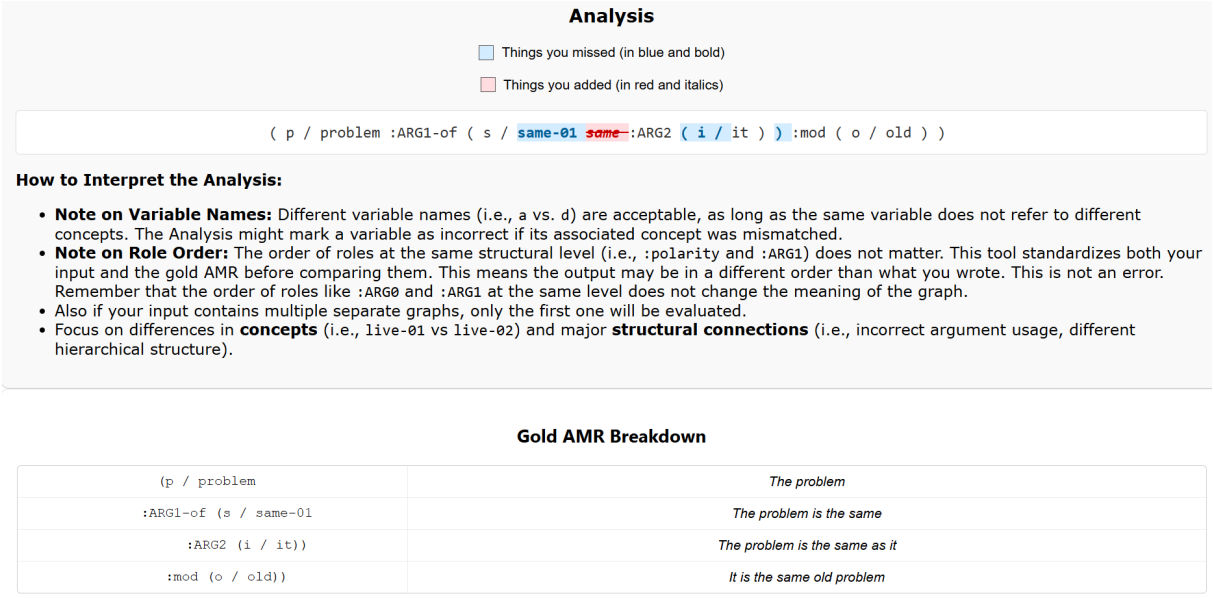

Figure 4: After selecting to show the Gold AMR in the Annotation Practice module, highlighted differences between the user-submitted and gold AMRs are shown as well as a natural-language interpretation of the gold AMR.

panel containing the textbox to submit the AMR annotation for the sentences, and the rightmost panel with options to view the gold AMR and the explanation of the contents of the gold AMR (see Figure 3 for an example). The explanation conceptually walks through the gold AMR graph's structure, highlighting the main concepts and roles in the context of the original sentence. We generate these explanations by hand for each of the 25 gold AMR graphs.

Finally, users can view a direct comparison between their submission and the gold AMR, which we show in a pane called Analysis at the bottom of the Annotation Practice page. Additionally, in the same pane, to aid users who are new to PENMAN notation, a Gold AMR Breakdown provides a natural-language explanation for each line of the gold AMR (Figure 4). This Gold AMR Breakdown is also generated by hand.

When the user input contains significant syntax errors, such as mismatched parentheses that make it difficult to parse the AMR, the submission is unable to be compared to the gold AMR via SMATCH. Thus, we show an error message to guide the user towards correcting the formatting of their input. Reasons for the AMR being invalid include the presence of mismatched parentheses, omitted or misspelled roles (such as forgetting `:` in front of `:ARGn`), and errors regarding concept variables (the letters that label concepts, such as `s` in `s / soul`). The most common variable errors are using duplicate variables for different concepts or omitting them entirely.

3.2.2. Backend

We make calls to the Google Sheets API in order to analyze user submissions.

The backend of the Analysis feature, which compares differences between the gold AMR and user submission, first checks for proper syntax by transforming the user input into PENMAN notation (Matthiessen and Bateman, 1991). Then, in order to verify that the order of roles is not mistakenly marked incorrect, we alphabetically sort all sibling role blocks (roles of the same structural level) in order, to easily find role matches between the submission and gold AMR. Then, we map the concept variable markers between the two graphs to ensure differences in variable naming are not counted as errors, since variable names are arbitrarily decided.[2] This mapping is executed sequentially without structural context, which is a naive approach that works efficiently for the purposes of providing feedback to users, but could be optimized to be more robust as a variant of SMATCH.

After normalization, we compare the tokenized strings of the two AMRs using the `jsdiff` library.[3] The resulting differences are then displayed in the following format: parts of the gold AMR that the user missed are shown in blue and bold, while the extra parts from the user's submission are showed in red and crossed-out italics. We note that using jsdiff is efficient for the sentence-level AMRs con-

[2]For example, `(f / funk)` and `(x / funk)` are equivalent AMRs though different variable markers are used for the `funk` concept.

[3]https://github.com/kpdecker/jsdiff

sidered here, but more scalable graph matching algorithms may be needed to handle much larger document-level graphs.

4. Pilot Studies

We conduct two pilot studies to evaluate the performance and effectiveness of TrAinMR. In particular, we set out to examine whether using TrAinMR leads to an improvement in a user's annotation quality, as measured by (1) SMATCH score against the gold reference, and (2) the user's perceived knowledge of AMR.

To answer these questions, each pilot study asks five annotators to complete the Tutorial and Annotation Practice modules of TrAinMR. We analyze their SMATCH scores against gold annotations and collect feedback on the user experience through post-study surveys.

4.1. Set-up

Each pilot study has five human annotators who are college students and are fluent in English, with little to no prior exposure to AMR, and zero annotation experience with AMR. The study instructions are as follows, provided orally:

- Read the Tutorial page
- Read the instructions at the top of the Annotation Practice page, then write AMR annotations for all 25 sentences
- Only one submission to each sentence is required, but we encourage additional attempts
- Rely on the Tutorial page and other resources linked from the page

After completing the task, annotators are also encouraged to fill out a feedback survey which includes a reflection on what aspects of TrAinMR they find most helpful during the annotation process, a quantitative rating of the utility of TrAinMR, and comments on what parts of the AMR annotation process they find particularly difficult.

In the first pilot study, we present the sentences in the Annotation Practice module in order of increasing complexity, as determined by the length of the sentence and the density of topics such as inverse relations and modals. Annotators also automatically see the gold AMR, explanation, breakdown, and highlighted differences after submitting their annotation. In this first pilot study, we do not track from which annotator each annotation is produced.

In the second pilot study, the study instructions remain the same as in the first pilot study. The study design is also kept the same, except for four logistical changes. First, the order of sentences is randomized per user, in order to assess whether annotator's abilities improve during the training process. The sentences are presented in random order generated using the Fisher-Yates shuffle algorithm (Durstenfeld, 1964). Second, the gold AMR and explanation are hidden by default and only appear after being selected by the user. Third, users are tracked using local storage. This means we can follow the trajectory of an individual user and analyze their performance. Finally, based on feedback from the first pilot study, an additional example is added to the inverse roles section in the Instructions, and reminders regarding parentheses-related syntax are added to the Tutorial and Annotation Practice pages.

4.2. SMATCH Score Results

In the first pilot study, the average SMATCH score (Cai and Knight, 2013) of all 210 annotations against gold standards is 0.669, with a standard deviation of 0.239. It is important to note that these scores exclude annotations that are unable to be evaluated by SMATCH. There are a total of 49 out of 210 annotations that produce this error. An extreme example is the sentence "He can't seem to help himself from apologizing for anything and everything", which only has two valid annotations that are used to compute the SMATCH score. The most common causes of parsing errors are mismatched parentheses and the usage of duplicate variable names for different concepts. For the first pilot study, we find that 16 out of the 25 sentences have an average SMATCH score between 0.600 and 0.800, and there is a small, negative correlation between sentence complexity and average SMATCH scores per sentence (R^2 = 0.198), suggesting that more complicated sentences are indeed harder to annotate.

Annotator ID	Average SMATCH	Variance
Annotator 1	0.434	0.012
Annotator 2	0.643	0.059
Annotator 3	0.497	0.036
Annotator 4	0.723	0.050
Annotator 5	0.645	0.033

Table 1: Average SMATCH and Variance per annotator in the second pilot study.

In the second pilot study, the average SMATCH scores (Cai and Knight, 2013) of all annotations against gold standards is 0.629, with a standard deviation of 0.229. The individual annotator scores and standard variance can be seen in Table 1.[4]

[4]Note that these scores exclude annotations that are unable to be parsed by SMATCH. There are a total of 80 out of 215 annotations that produce this error. The most common reason for this error is the presence of duplicate variable nodes.

Each annotator is assigned a unique, anonymous identifier (Annotator 1, Annotator 2, etc.) that is used consistently throughout the analysis. We see that Annotator 1 has the most consistent performance, as they have the lowest variance, while Annotator 2 exhibits the most varied performance. Individuals learn at different speeds and gain varying proficiency levels. Performing individual analyses reflects this reality and provides a guide for improving the training schema. For example, this data could advocate for an adaptive learning system that provides supplementary practice to annotators with lower scores, while providing more challenging examples to those who appear to learn more quickly.

Given that the second pilot study shows the sentences to annotators in a randomized order, we assess whether the average SMATCH score against the gold reference increases as the annotators practice more. We measure learning (as seen in Figure 5) by viewing annotator performance in the order of sentences that they annotated. In this second pilot study, Annotators 1-4 show a positive correlation between SMATCH score and annotations made over time. Annotator 1 and Annotator 4 show the highest correlation, with an R^2 value of 0.276 and 0.315 respectively, suggesting improvement in AMR annotation via TrAinMR. In contrast to the other annotators, Annotator 5 is able to achieve high SMATCH scores since the beginning of their annotation process, suggesting their learning from the Tutorial page. As a result, Annotator 5's performance shows a weak negative correlation (R^2 value is less than 0.1). We note that these R^2 values are from a simple linear fit and serve as a descriptive trend. Since we do not explicitly model variations in sentence difficulty nor assume a linear learning pattern, we place more emphasis on interpreting these results qualitatively.

The average SMATCH scores across all annotations between the first and second pilot study are relatively similar (0.669 and 0.629 respectively). These SMATCH scores indicate that annotators without previous exposure to AMR can produce annotations that show moderate overlap with gold-standard graphs after use of TrAinMR. While these performance scores are lower than the inter-annotator agreement score ranging from 0.71 to 0.89 SMATCH (Wein, 2025), they suggest that TrAinMR has the potential to help provide foundational knowledge for complex annotation tasks. However, this interpretation is limited given the small pilot size and number of invalid annotations.

4.3. Feedback Survey Results and Error Analysis

In the feedback survey with nine responses (one annotator in the first study did not respond to the form, hence nine total responses instead of ten), annotators report their prior knowledge of AMR annotation on a scale of 1 (None) to 5 (Expert). The responses show that most annotators are novices, with a mode of 1 and a mean score of 1.550. Eight out of nine participants rate their prior exposure as a 1 or a 2, and one rates their proficiency as a 3. When asked if TrAinMR helped them learn AMR annotation (1=Not at all, 5=Very much), the feedback is positive. The mode is 4 and the mean score is 3.330. This indicates that TrAinMR can help expose users to AMR in an interactive way, compiling the most necessary information to get one started in AMR annotation.

Annotators also note that time spent reading the Tutorial lasted approximately 20-30 minutes for each user. The fastest completion time for annotating the 25 sentences is approximately 1 hour and 13 minutes and the longest completion time is approximately 2 hours and 12 minutes. One participant in particular notes that the annotations took less time as they completed more practice.

Annotators state that they have the most trouble with finding the root of the sentence, distinguishing between roles and relations with similar uses, understanding inverse roles, and finding the correct PropBank frames. For example, for the sentence "In recently years, Finland has been keeping an obvious trade surplus in this region," we see that the annotators have difficulty identifying the correct concepts, including distinguishing when to use a PropBank frame or a word (`obvious` v.s. `obvious-01`).[5]

We identify several other common errors in the annotations. The sentence with the lowest average SMATCH score in the first pilot study is "The Japanese delegation will fly to Beijing on the 2nd" with SMATCH of 0.415. Although there are five annotators in this study, they are encouraged to resubmit their work, resulting in a total of eight submissions for this particular sentence. Of these eight submissions, four produce parsing errors and are thus excluded from the average score calculation. Out of the valid submissions, annotators appear to have trouble with the exact syntax of the `:name` role and wikification. Annotators also show difficulty in identifying the correct role for the locations in the sentence, mixing between `:destination`, `:mod`, and `:ARG1-of`.

The sentence with the lowest average SMATCH

[5]Note that the the original sentence contains "recently" though "recent" would be grammatical; we present the sentence as written to the annotator.

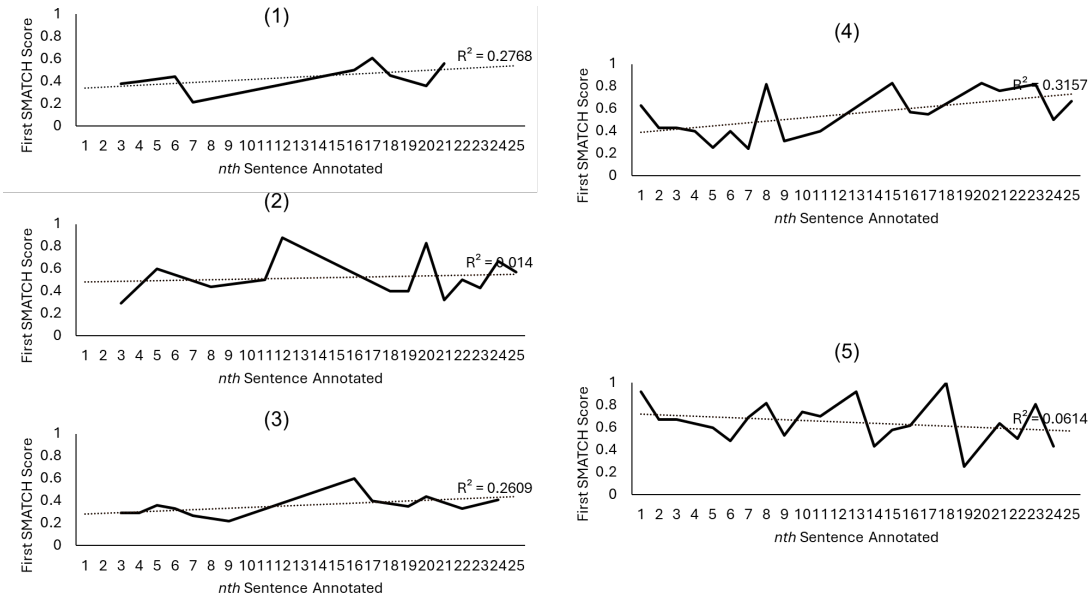

Figure 5: The performance of each annotator from the second pilot study is shown over time. The x-axis represents the nth sentence annotated, and the y-axis represents the SMATCH score of the first annotation submission per sentence. The first submission for each sentence per annotator is being compared. Missing values are included. Annotators 1-5 are represented by the labeled graphs.

score in the second pilot study is "a group of people of nine professions" with a SMATCH score of 0.373. Two out of five annotations produce parsing errors for this sentence, and thus are excluded from the score. The annotators seem to struggle with using reifications, where none of the annotators use `have-org-role-91` in their submission for this sentence.

Our pilot studies also reveal annotators' trouble in identifying the correct root concept. In regards to the first study, in the sentence "The policy is a matter of national sovereignty and security," six out of eight submissions exclude the relation `:domain`. Half of the submissions correctly identify `matter` as the root concept, of which two of them are incorrectly `matter-01` and `matter-02`. The other submissions have `policy` and `be-03` as the root concept. There are similar patterns of annotators confusing PropBank frames with regular words, such as in "During a time of prosperity and happiness, such a big earthquake suddenly struck," where the concept `happiness` in the gold AMR is written as `happy-01` and `happy-02` in two of the submissions. We also note that annotators seem to forget to add demonstrative pronouns sometimes such as "this," as seen in submissions for the sentence "He felt that, there were more new competitors from our country participating in this competition." In the second study, another example of annotators having a difficult time identifying the root concept is for the sentence "The acquisition is expected to be completed before April," where the root concept is `expect-01`. Eight out of ten submissions correctly identify this, however one annotator's initial submission uses `complete-01` as the root instead.

Another key finding from our analysis is related to concept selection. For instance, for one sentence, six out of nine submissions in the second study use the adjective `happy` while the gold AMR uses the noun `happiness`. While the annotators successfully identify the main semantic concept in this case, they struggle with nominalization. This suggests that our training is effective at conveying the main principles of AMR, but specific linguistic details like nominalization is a challenge for novices. This insight informs how to improve future versions of the training tutorial.

Perhaps unsurprisingly, we find annotators are unable to identify specific roles or relations that are not included in the Tutorial. For example, in the sentence "Why is it so hard to understand?", the gold AMR uses `:degree`, a relation that is not included in the Tutorial, for the concept `so`. Two out of the ten annotations (five being resubmissions) in the second study include the concept `so`. Annotator 1 connects the concept `so` to the relation `:manner` as their first submission. Annotator 5 connects the concept `so` to the relation `:quant` in their resubmission, whereas they omit `so` in their initial submission. The other annotators omit the concept `so` and `:quant` in their submissions.

Finally, when asked which resources users find most helpful, annotators could select multiple options. The responses indicate that a combination of resources is valuable for the learning process. Seven out of nine total responses across both stud-

ies indicate that the Tutorial page is most helpful, and six out of nine responses state that the explanation of the gold AMR is most helpful. External resources such as referencing PropBank frames and AMR guidelines, and the gold AMR itself, are also indicated as most helpful by five respondents. This indicates that users find TrAinMR particularly effective when supplemented with examples and further documentation.

5. Conclusion

In this paper, we present TrAinMR, an open-source, web-based tool designed to address the difficulty in creating large-scale AMR corpora due to the time and expertise needed to train new annotators. TrAinMR provides a foundational tutorial and an interactive practice environment that can be reused as a training schema. Its immediate feedback loop allows novices to compare their attempts to a gold standard AMR, helping them to learn the accurate application of AMR's guidelines across annotations. By having a self-contained learning module with original examples and explanations, TrAinMR makes the complex task of AMR annotation training more accessible to a broader audience.

We also review the results to two pilot studies evaluating how effective TrAinMR is. We find that many annotators make two common mistakes, which are missing or mismatching parentheses, and using the same variable label for different concepts. Other common mistakes include using incorrect PropBank frames and different roles and relations.

The average SMATCH score across all annotations in pilot study one and two are 0.669 and 0.629 respectively, which provides an initial indication of the potential utility of our training tool. We observe a trend in four annotators in the second pilot study where their performance is positively correlated with the number of annotations completed. Additionally, another annotator shows an increase in performance on complex sentences, which may suggest they become more proficient at writing annotations during their continued use of TrAinMR.

Common error patterns in the two pilot studies also highlight the importance of mitigating the effects of annotator subjectivity. Annotators may disagree with certain parts of an AMR graph depending on their own interpretation of words, such as finding a different verb in a sentence to be the root of an AMR, which can also be due to language ambiguity (Wein, 2025). We see this in the user annotations in our study, where places like the root concept and certain roles are different from the gold AMR, but the overall meaning of the AMRs are not dissimilar.

Future work may include enhancing TrAinMR with an LLM-based feedback feature to provide more interactive, natural language explanations of differences between a user's submission and the gold AMR for a large set of sentences. Our error analysis also informs the development of other training curriculum. If additional documentation on approaches to AMR training emerges, TrAinMR could be compared with other training approaches in future work. Finally, expansions to TrAinMR could support AMR extensions and other languages in order to make semantic annotation beyond AMR more accessible.

Limitations

Our study is limited to AMR annotation of texts in the English language, which may not generalize to other languages or annotation frameworks. Additionally, our sample size is reduced due to some submitted annotations being invalid AMR graphs, meaning we are unable to calculate SMATCH scores with them. We use pilot studies to iteratively improve and validate the utility of our tool; future pilot studies may examine how annotators trained using TrAinMR may perform over a baseline, self-training for the same amount of time, or assess how whether TrAinMR enables improved quality control of annotations.

Acknowledgments

We thank anonymous reviewers and members of the Amherst College NLP lab for their feedback. This work is supported by the Amherst College HPC, which is funded by NSF Award 2117377.

6. Bibliographical References

Shafiuddin Rehan Ahmed, Jon Cai, Martha Palmer, and James H. Martin. 2024. X-AMR annotation tool. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*, pages 177–186, St. Julians, Malta. 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.

Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and

Nathan Schneider. 2019. Abstract meaning representation (amr) 1.2.6 specification.

Julia Bonn, Martha Palmer, Zheng Cai, and Kristin Wright-Bettner. 2020. Spatial AMR: Expanded spatial annotation in the context of a grounded Minecraft corpus. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 4883–4892, Marseille, France. European Language Resources Association.

Richard Brutti, Lucia Donatelli, Kenneth Lai, and James Pustejovsky. 2022. Abstract Meaning Representation for gesture. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 1576–1583, Marseille, France. European Language Resources Association.

Jon Cai, Shafiuddin Rehan Ahmed, Julia Bonn, Kristin Wright-Bettner, Martha Palmer, and James H. Martin. 2023. CAMRA: Copilot for AMR annotation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 381–388, Singapore. Association for Computational Linguistics.

Jon Cai, Kristin Wright-Bettner, Zekun Zhao, Shafiuddin Rehan Ahmed, Abijith Trichur Ramachandran, Jeffrey Flanigan, Martha Palmer, and James Martin. 2025. LiDARR: Linking document AMRs with referents resolvers. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, pages 426–435, Vienna, Austria. 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.

Wei-Te Chen and Will Styler. 2013. Anafora: A web-based general purpose annotation tool. In *Proceedings of the 2013 NAACL HLT Demonstration Session*, pages 14–19, Atlanta, Georgia. Association for Computational Linguistics.

Richard Durstenfeld. 1964. Algorithm 235: random permutation. *Communications of the ACM*, 7(7):420.

Allyson Ettinger, Jena Hwang, Valentina Pyatkin, Chandra Bhagavatula, and Yejin Choi. 2023. "you are an expert linguistic annotator": Limits of LLMs as analyzers of Abstract Meaning Representation. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 8250–8263, Singapore. Association for Computational Linguistics.

Ulf Hermjakob. 2013. Amr editor: A tool to build abstract meaning representations. *Marina del Rey, CA. USC Information Sciences Institute*.

Kevin Knight, Bianca Badarau, Laura Banarescu, Claire Bonial, Madalina Bardocz, Kira Griffitt, Ulf Hermjakob, Daniel Marcu, Martha Palmer, Tim O'Gorman, et al. 2020. Abstract Meaning Representation (AMR) Annotation Release 3.0. Technical Report LDC2020T02, Linguistic Data Consortium, Philadelphia, PA.

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.

Mary Martin, Cecilia Mauceri, Martha Palmer, and Christoffer Heckman. 2020. Leveraging non-specialists for accurate and time efficient amr annotation. In *Proceedings of the LREC 2020 Workshop on" Citizen Linguistics in Language Resource Development"*, pages 35–39.

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.

Christian Matthiessen and John A Bateman. 1991. *Text generation and systemic-functional linguistics: experiences from English and Japanese*. Pinter Publishers.

Martha Palmer, Daniel Gildea, and Paul Kingsbury. 2005. The Proposition Bank: An annotated corpus of semantic roles. *Computational Linguistics*, 31(1):71–106.

Nathan Schneider, Tim O'Gorman, and Jeffrey Flanigan. 2015. The logic of amr practical, unified, graph-based sentence semantics for nlp.

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.

Jens EL Van Gysel, Meagan Vigus, Jayeol Chun, Kenneth Lai, Sarah Moeller, Jiarui Yao, Tim

O'Gorman, Andrew Cowell, William Croft, Chu-Ren Huang, et al. 2021. Designing a uniform meaning representation for natural language processing. *KI-Künstliche Intelligenz*, 35(3):343–360.

Shira Wein. 2025. Ambiguity and disagreement in Abstract Meaning Representation. In *Proceedings of Context and Meaning: Navigating Disagreements in NLP Annotation*, pages 145–154, Abu Dhabi, UAE. International Committee on Computational Linguistics.

Shira Wein and Nathan Schneider. 2024. Assessing the cross-linguistic utility of abstract meaning representation. *Computational Linguistics*, 50(2):419–473.