REMATCH: Robust and Efficient Knowledge Graph Matching for Improved Structural and Semantic Similarity

Anonymous ACL submission

Abstract

Knowledge graphs play a pivotal role in various applications, such as question-answering and fact-checking. Abstract Meaning Representation (AMR) represents text as knowledge graphs. Evaluating the quality of these graphs involves matching them structurally to each other and semantically to the source text. Existing AMR metrics are inefficient and struggle to capture semantic similarity. We also lack a systematic evaluation benchmark for assessing structural similarity between AMR graphs. To overcome these limitations, we introduce a novel AMR similarity metric, *rematch*, alongside a new evaluation for structural similarity called RARE. Among state-of-the-art metrics, *rematch* ranks second in structural similarity; and first in semantic similarity by 1–5 percentage points on the STS-B and SICK-R benchmarks. *Rematch* is also five times faster than the next most efficient metric.

1 Introduction

Knowledge graphs provide a powerful framework for multi-hop reasoning tasks, such as question answering and fact-checking (Yasunaga et al., 2021; Vedula and Parthasarathy, 2021). Even for closed-domain tasks like long-form question answering and multi-document summarization, knowledge graphs derived from documents exhibit superior performance compared to plain text (Fan et al., 2019). This highlights the significance of automatically parsed knowledge graphs in both large-scale and fine-grained structured reasoning applications.

The Abstract Meaning Representation (AMR) framework leverages acyclic, directed, labeled graphs to represent semantic meaning (knowledge) extracted from text (Banarescu et al., 2013). As illustrated in the example of Fig. 1, AMRs capture the relationships between concepts and their roles in a sentence. They have been applied to a variety of natural language processing tasks, including summarization and question answering (Liu et al., 2015; Hardy and Vlachos, 2018; Bonial et al., 2020; Mitra and Baral, 2016). Recent work has also shown that AMRs can reduce hallucinations and improve performance in factual summarization tasks (Ribeiro et al., 2022).

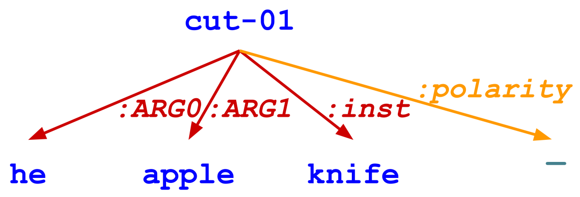

Figure 1: AMR for the sentence: *"He did not cut the apple with a knife."* Colors indicate AMR components: instances (blue), relations (red), constants (teal), and attributes (orange). The instance **cut-01** is a *PropBank* frame that uses ***ARG0***, ***ARG1*** and ***inst*** to express the verb's agent (**he**), patient (**apple**), and instrument (**knife**), respectively. The attribute ***polarity*** expresses the negation of the verb through the constant **–**.

However, evaluating the quality of knowledge graphs like AMRs hinges critically on the ability to accurately measure similarity. This assessment must consider a dual perspective. Firstly, the similarity between two AMRs should reflect structural consistency, guaranteeing that the similarity between two AMRs aligns with the similarity of their structural connections. Secondly, AMRs should exhibit semantic consistency, ensuring that the similarity between two AMRs aligns with the similarity of the texts from which they are derived. Therefore, an effective AMR similarity metric must successfully account for both structural and semantic similarity, all while overcoming the resource-intensive nature of matching labeled graphs.

Current AMR similarity metrics fall short in several key areas. Firstly, their computational efficiency hinders the comparison of large AMRs extracted from documents (Naseem et al., 2022). Secondly, these metrics struggle to accurately capture the semantic similarity of the underlying text from which AMRs are derived (Leung et al., 2022). Additionally, while recent efforts like BAMBOO (Opitz et al., 2021) have evaluated metrics on AMR transformations, we still lack a large-scale benchmark to systematically evaluate the ability of AMR metrics to capture structural similarity.

Our work introduces a structural AMR benchmark called *Randomized AMRs with Rewired Edges* (RARE) and proposes *rematch*, a novel and efficient AMR similarity metric that captures both structural and semantic similarity. Compared to the state of the art, *rematch* trails the best similarity metric on RARE by 1 percentage point and ranks first on the STS-B (Agirre et al., 2016) and SICK-R (Marelli et al., 2014) benchmarks by 1–5 percentage points. Additionally, *rematch* is five times faster than the next most efficient metric.

2 Background

2.1 Abstract Meaning Representations

Abstract Meaning Representation (AMR) is a structural, explicit language model that utilizes directed, labeled graphs to capture the semantics of text (Banarescu et al., 2013). AMR is designed to be independent of surface syntax, ensuring that sentences with equivalent meanings are represented by the same graph. An AMR comprises three fundamental components: instances, attributes, and relations.

1. **Instances** are the core semantic concepts. Structurally, they are represented by nodes in the graph. AMRs have two types of instances. One utilizes *PropBank* (Palmer et al., 2005), a dictionary of frames that map verbs and adjectives. The other comprises entities. Considering the sentence in Fig. 1, *"He did not cut the apple with a knife,"* the AMR contains a *PropBank* instance **`cut-01`** and three entity instances: **`he`**, **`apple`** and **`knife`**.

2. **Attributes** capture details about instances, such as names, numbers, and dates. These values are represented as constant nodes. Structurally, an attribute is identified in the graph as the edge from an instance node to a constant node. For example, in Fig. 1, the attribute ***`polarity`*** is specified for the instance **`cut-01`**, where **`-`** is the constant that represents the negation of the verb.

3. **Relations** represent the connections between instances. In Fig. 1, the instance **`cut-01`** has three outgoing relations: ***`ARG0`***, ***`ARG1`***, and ***`inst`***. These come from *PropBank*'s **`cut-01`** frame and link to the agent (**`he`**), the patient (**`apple`**), and the instrument (**`knife`**), respectively.

2.2 AMR Similarity

Graph isomorphism is a test to determine whether two graphs are structurally equivalent. The class-wise isomorphism testing with limited backtracking (CISC) algorithm efficiently identifies isomorphic relationships in labeled graphs (Hsieh et al., 2006), such as AMRs. But a pair of AMRs may not have the same number of nodes, which violates a key assumption of graph isomorphism. A more appropriate approach is subgraph isomorphism, which determines whether a smaller graph is isomorphic to a subgraph of a larger graph. Subgraphs of directed acyclic graphs, like AMRs, can be enumerated in polynomial time (Peng et al., 2018), enabling efficient application of the CISC test to each pair of smaller AMR and larger AMR subgraphs. However, even if two AMRs are not subgraph-isomorphic, they may still exhibit similarities in meaning and structure. Next, we describe various existing approaches to measure the similarity between AMR graphs.

2.3 AMR Similarity Metrics

2.3.1 Smatch

Smatch is a prominent tool for evaluating AMR parsers (Cai and Knight, 2013). It establishes AMR alignment by generating a one-to-one node mapping, considering node and edge labels. To efficiently explore this vast mapping space, *smatch* employs a hill-climbing heuristic.

2.3.2 S2match

Similar to *smatch*, *s2match* (Opitz et al., 2020) also establishes a node alignment between two AMRs. However, instead of relying on AMR labels, *s2match* utilizes GloVe word embeddings (Pennington et al., 2014). To address the extensive search space, it uses the same hill-climbing heuristic adopted by *smatch*.

2.3.3 Sembleu

Sembleu generates path-based n-grams from AMRs by leveraging node and edge labels (Song and Gildea, 2019). The final similarity score for an AMR pair is determined by calculating the BLEU score (Papineni et al., 2002) between their n-grams. By avoiding a one-to-one node alignment, *Sembleu* efficiently bypasses the issue of exploring a large search space.

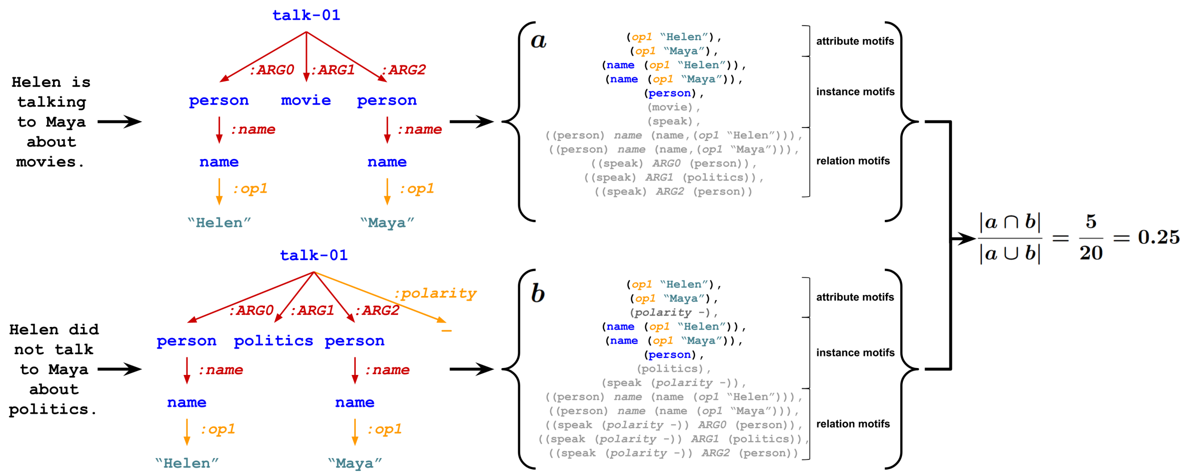
Helen is talking to Maya about movies.
talk-01
:ARG0
:ARG1
:ARG2
person
movie
person
:name
name
:op1
"Helen"
"Maya"
a
(op1 "Helen"),
(op1 "Maya"),
(name (op1 "Helen")),
(name (op1 "Maya")),
(person),
(movie),
(speak),
((person) name (name,(op1 "Helen"))),
((person) name (name,(op1 "Maya"))),
((speak) ARG0 (person)),
((speak) ARG1 (politics)),
((speak) ARG2 (person))
attribute motifs
instance motifs
relation motifs
Helen did not talk to Maya about politics.
:polarity
politics
b
(op1 "Helen"),
(op1 "Maya"),
(polarity -),
(name (op1 "Helen")),
(name (op1 "Maya")),
(person),
(politics),
(speak (polarity -)),
((person) name (name (op1 "Helen"))),
((person) name (name (op1 "Maya"))),
((speak (polarity -)) ARG0 (person)),
((speak (polarity -)) ARG1 (politics)),
((speak (polarity -)) ARG2 (person))
$\frac{|a \cap b|}{|a \cup b|} = \frac{5}{20} = 0.25$

Figure 2: An example of *rematch*'s similarity calculation for a pair of AMRs. After AMRs are parsed from sentences, *rematch* has a two-step process to calculate similarity. First, sets of motifs are generated. Second, the two sets are used to calculate the Jaccard similarity (intersecting motifs shown in color).

2.3.4 WLK and WWLK

The Weisfeiler-Leman Kernel (*WLK*) and Wasserstein Weisfeiler-Leman Kernel (*WWLK*) for AMRs also utilize graph features for computing similarity (Opitz et al., 2021). *WLK* first constructs node features by recursively aggregating AMR node and edge labels. Then it generates a frequency-based feature vector for each AMR and calculates a similarity score using their inner product. *WWLK* extends *WLK* with features based on aggregated node embeddings (GloVE) instead of node labels.

3 Methods

In this work, we propose *rematch*, an AMR similarity metric that aims to capture both the structural and semantic overlap between two AMRs.

3.1 Rematch

A straightforward approach to match two labeled graphs involves identifying the alignment between node labels. However, labeled graphs often contain duplicate labels, necessitating an exhaustive exploration of all one-to-one combinations among nodes within the same label group to determine the optimal match. The resulting matching complexity hinges on the size of node groups with shared labels. This is why algorithms like *smatch* and *s2match* do not scale well to large AMRs, where these node groups can be large.

Graph features constructed using an ordered concatenation of edge-node bi-grams are utilized in both isomorphism tests like the CISC and kernels like Weisfeiler-Leman (Shervashidze et al., 2011). This approach is effective: it consistently produces smaller node groups compared to those based solely on node labels. Matching between two graphs is significantly accelerated as a result.

Inspired by this idea of exploiting graph features for efficiency, *rematch* computes the similarity between two AMRs by analyzing the overlap of semantically rich features, which we call *motifs*. Unlike the ordered graph partitions used by CISC and Weisfeiler-Leman Kernel, which rely on node and edge labels, AMR *motifs* are unordered graph partitions that leverage AMR instances, attributes, and relations. This approach allows *rematch* to capture meaning across three semantic levels: specific facts (attributes), main concepts (instances), and the relationships among concepts (relations). Fig. 2 illustrates *rematch* through an example. Next, we delve into the three orders of semantic motifs that we use for *rematch*. We extract these motifs using the Python package Penman (Goodman, 2020).

1. **Attribute motifs** are pairs of attributes and constants associated with AMR instance nodes. For the bottom AMR in Fig. 2, `talk-01` has attribute motif (`polarity -`), indicating a negation. The first `name` has the attribute motif (`op1 "Helen"`) and the second `name` has (`op1 "Maya"`), identifying the name values. The remaining instances do not have any attributes.

2. **Instance motifs** leverage *Verbatlas*, a resource that maps *PropBank* frames to more generalized frames (Di Fabio et al., 2019). If an instance in the AMR corresponds to a Verbatlas frame, the

latter is used instead. Otherwise, the original *PropBank* instance is retained. For example, in Fig. 2, `talk-01` is replaced by the more generalized Verbatlas frame `speak`. The generation of instance motifs follows two approaches. If an instance lacks associated attributes, the instance itself serves as its motif. However, if attributes are present, instance motifs are constructed by combining the instance with each of its attribute motifs. For the bottom AMR in Fig. 2, the instance motif for `talk-01` is (`speak` (`polarity` `-`)), indicating a negation of the verb. For the two `person` instances and the `politics` instance, the instances themselves become their motifs, namely (`person`) and (`politics`). Finally, the instance motifs for the two `name` instances are (`name` (`op1` `"Helen"`)) and (`name` (`op1` `"Maya"`)) respectively, identifying the names in the conversation.

3. **Relation motifs** are constructed for relation edges in an AMR graph. Each relation motif comprises three elements: an instance motif of the source instance, the relation label, and an instance motif of the target instance. A relation can have multiple relation motifs, one for each unique combination of source and target instance motifs. For the bottom AMR in Fig. 2, the relation motifs for `ARG0`, `ARG1` and `ARG2` are: ((`speak` (`polarity` `-`)) `ARG0` `person`), indicating a person is the speaker of the conversation; ((`speak` (`polarity` `-`)) `ARG1` `politics`), indicating that the topic of conversation is politics; and ((`speak` (`polarity` `-`)) `ARG2` `person`), indicating that a person is the recipient of the conversation. For the two `name` relations, the motifs are: ((`person`) `name` (`op1` `"Helen"`))), identifying "Helen" as the name of one person; and ((`person`) `name` (`op1` `"Maya"`)), identifying "Maya" as the name of the other person.

Each AMR is represented by the union of its instance, relation, and attribute motifs. The *rematch* score between two AMRs is determined by calculating the Jaccard similarity between their respective motif sets, as illustrated in Fig. 2.

4 Evaluation

We evaluate the effectiveness of *rematch* on three types of similarity: structural similarity, semantic similarity, and BAMBOO (Opitz et al., 2021), a hybrid benchmark that modifies AMR semantics through structural transformations. Additionally, we assess the efficiency of *rematch*.

4.1 Structural Similarity (RARE)

Given that AMRs are graphical representations of text, an AMR similarity metric should be sensitive to structural variations between AMRs, even if its labels remain unchanged.

Since there is no established evaluation of AMR metrics on structural similarity, we have developed a new benchmark dataset called *Randomized AMRs with Rewired Edges* (RARE). RARE consists of English AMR pairs with similarity scores that reflect the structural differences between them.

In the construction of RARE, we adopt the iterative randomization technique commonly used for graph rewiring. This involves repeatedly selecting a random pair of directed edges and swapping either their source or target nodes to establish new connections. This way each node's in-degree and out-degree are preserved. In applying this approach to AMRs, we swap a random pair of edges between either attributes or relations. This allows us to quantify the structural changes made to the AMR through the number of swapped edges. We generate a spectrum of modified graphs from an original AMR, ranging from the unchanged graph to one where all edges are rewired, subject to some constraints that preserve the integrity of AMRs:

1. **Structural Constraints.** AMRs are acyclic, connected graphs that allow no multiedges (more than one edge between the same pair of nodes). To preserve these properties during the rewiring process, pairs of swapped edges must maintain these constraints in the modified AMR.

2. **Semantic Contraints.** These contraints relate to swapping attributes and relations:

   (a) Attributes have an inherent connection with constants in AMRs. Hence, while rewiring a pair of attribute edges, only the source instance node should be swapped. This restriction ensures that the association between the attribute and its corresponding constant remains intact. For example, the constant node `-` should remain associated solely with the attribute edge `polarity`.

   (b) Relations in AMRs connect two instances. When rewiring a pair of relation edges, only the target instance node should be swapped. This restriction maintains the association between the relation's source instance and the relation itself. For example, *PropBank* instances have a predefined set of relations with which they can be associated. The instance node `talk-01` can only be associated with edges ***ARG0***, ***ARG1***, and ***ARG2***.

Each pair of AMRs, consisting of an original AMR G with E edges and its corresponding rewired AMR G' with E' swapped edges, is annotated with the following similarity score:

$$similarity(G, G') = \frac{|E| - |E'|}{|E|} \quad (1)$$

To generate the RARE benchmark, we licensed the English AMR Annotation 3.0 (Knight, Kevin et al., 2020) containing 59,255 human-created AMRs. Using the process described above results in 563,143 rewired AMR pairs annotated with similarity scores per Eq. 1. RARE is derived from AMR 3.0 by merging, shuffling, and re-splitting the original data into training (47,404), development (5,925), and test (5,926) sets, following an 80-10-10 split ratio. The corresponding counts in these splits are 450,067, 56,358, and 56,718, respectively. The creation of training and development splits could facilitate the future development of supervised AMR metrics. For the current evaluation, AMR structural similarity metrics are evaluated on the RARE test split.

We evaluate a similarity metric by computing the Spearman correlation between its scores and the ground truth values from Eq. 1, across a set of pairs of original and modified AMRs. We refer to this as the *structural consistency* of the metric.

4.2 Semantic Similarity

A fundamental tenet of AMRs is that if two pieces of text are semantically related, their corresponding AMRs should exhibit a degree of similarity. Based on this assumption, we evaluate an AMR similarity metric by considering many pairs of sentences. For each pair, we compare the similarity generated by the metric for the corresponding AMRs to a human-annotated similarity score between the sentences.

We utilize two standard sentence similarity benchmarks for English: STS-B (Agirre et al., 2016) and SICK-R (Marelli et al., 2014). To account for variations in AMR parsing accuracy, we employ four different AMR parsers: *spring* (Bevilacqua et al., 2021), *amrbart* (Bai et al., 2022), *structbart* (Drozdov et al., 2022), and the *maximum Bayes smatch ensemble* (Lee et al., 2022).

Given a set of sentence pairs and corresponding AMR pairs, we evaluate a similarity metric by computing the Spearman correlation between its scores for the AMR pairs and the human-annotated similarity values for the sentence pairs. We refer to this as the *semantic consistency* of the metric. Note that semantic consistency can be used to evaluate any similarity method for sentences, not only AMR-based ones. For both structural and semantic consistency, we use Spearman rather than Pearson correlation because we do not assume that the similarity values are normally distributed.

4.3 Hybrid Similarity (BAMBOO)

In addition to the structural and semantic consistency discussed earlier, we evaluate the robustness of AMR metrics using the *Benchmark for AMR Metrics Based on Overt Objectives*, or BAMBOO (Opitz et al., 2021). BAMBOO assesses the ability of AMR similarity metrics to capture semantic similarity between English sentences while modifying the structure of the corresponding AMRs.

BAMBOO incorporates three types of graph modifications: synonym replacement, reification, and role confusion. Consider the example sentence "He lives in the attic," represented by an AMR where the node `live-01` connects to nodes `he` and `attic` via the edges ***ARG0*** and ***location***, respectively. Synonym replacement swaps *PropBank* instances with equivalent terms. In the example, `live-01` might be replaced by `reside-01`. Reification transforms a relation into a new instance. In the example, the ***location*** edge might be replaced by a new node `be-located-at-91` connected to `live-01` and `attic` via new ***ARG1*** and ***ARG2*** edges, respectively. Finally, role confusion swaps relation roles. In the example, the relations ***location*** and ***ARG0*** might be swapped such that the modified AMR would represent the sentence "The attic lives in him." BAMBOO applies these modifications to the original train, test and dev splits of the STS-B, SICK-R, and PARA (Dolan and Brockett, 2005) datasets.

Given a set of modified AMR pairs, BAMBOO evaluates an AMR metric by the Spearman correlation[1] between its scores and the similarity between the corresponding sentence pairs. We call this *hybrid consistency* of the metric.

[1]The original formulation of BAMBOO (Opitz et al., 2021) used Pearson correlation. Here we use Spearman because, as for structural and semantic consistency, we do not assume that the similarity values are normally distributed.

AMR Metric	RARE
smatch	**96.57**
s2match	94.11
sembleu	94.83
WLK	90.39
WWLK	86.31
rematch	95.32

Table 1: Structural consistency of different AMR similarity metrics on the RARE test split.

4.4 Efficiency

As discussed earlier, the computational complexity associated with node alignment is a crucial challenge for comparing AMRs. To address this issue, we evaluate the search spaces explored by various metrics and the required runtime.

We establish a realistic test bed using the AMR Annotation 3.0 once again. For this evaluation, we randomly sampled 500,000 pairs from the $\binom{59,255}{2}$ possible AMR combinations. For each pair of AMRs (G_1, G_2), the search spaces for node alignment algorithms like *smatch* and *s2match* is

$$search(G_1, G_2) = \prod_{n_i \in G_1} |M_{G_2}(n_i)| \qquad (2)$$

where $M_{G_2}(n_i)$ denotes the set of matching candidates in G_2 for node n_i. For feature-based algorithms, like *sembleu*, *WLK*, and *rematch*, we record the search space using

$$search(G_1, G_2) = |\mathcal{F}(G_1)| \cdot |\mathcal{F}(G_2)| \qquad (3)$$

where $\mathcal{F}(G)$ denotes the feature set for graph G. For each pair of AMRs, we also record the runtime. We could not record the search space or runtime for *WWLK* given its batch-style execution.

5 Results

5.1 Structural Consistency

Table 1 reports on the structural consistency of the AMR similarity metrics on the RARE test split. We can see that *smatch* performs the best, followed closely by *rematch*, *sembleu* and *s2match*. The subpar performance of *WLK* and *WWLK* can be attributed to their reliance on features using all of a node's neighbors. This approach results in changes to node features regardless of the number of modified neighbors, failing to capture the nuances of neighborhood changes.

5.2 Semantic Consistency

Table 2 reports on the semantic consistency of the similarity metrics for different AMR parsers. *Rematch* outperforms all other metrics by 1–5 percentage points, across all parsers and benchmarks. The *mbse* and *amrbart* parsers perform best for the STS-B and SICK-R datasets, respectively.

So far we have focused on methods that use AMRs to calculate the semantic similarity between sentences. Table 3 reports on the evaluation of alternative similarity methods on the same benchmarks. Like AMR-based methods, these are also unsupervised (not trained specifically) for textual semantic similarity. AMR outperforms some representations like GloVe and RoBERTa but lags behind the state-of-the-art method SimCSE (Gao et al., 2022).

5.3 Hybrid Consistency

Table 4 reports on the hybrid consistency of AMR similarity metrics on the four different tests of BAMBOO, across three different datasets. The results vary considerably across graph modifications and datasets; none of the methods is a clear winner. *Rematch* achieves best results in three out of twelve tests and lags slightly behind *s2match* on average.

5.4 Efficiency

Fig. 3 shows the search spaces explored by AMR metrics for increasing values of N, the average size of each pair of AMRs. The size of each AMR is determined by the sum of the number of instances, attributes, and relations. Approaches that find node alignment between AMRs, like *smatch* and *s2match*, explore search spaces that grow exponentially with N. Feature-based methods, like *sembleu*, *WLK*, and *rematch*, in contrast, explore significantly smaller spaces.

Fig. 3 also shows the runtimes for increasing N. By using a hill-climbing heuristic, node-alignment metrics effectively overcome the exponentially growing search spaces. However, they are significantly less efficient compared to feature-based metrics. For large values of N, *smatch* and *s2match* display an approximately quadratic time complexity. *Sembleu*, *WLK*, and *rematch*, on the other hand, demonstrate a linear complexity.

In terms of absolute runtime on the test bed, *rematch* is the fastest metric, with a runtime of 51 seconds. This is five times faster than *sembleu*, which took 275 seconds. *Smatch*, *s2match*, and

	STS-B				SICK-R			
	spring	*amrbart*	*sbart*	*mbse*	*spring*	*amrbart*	*sbart*	*mbse*
smatch	53.84	54.67	54.73	55.16	58.69	58.89	58.70	57.84
s2match	56.60	57.15	57.54	57.64	58.09	58.56	58.42	57.58
sembleu	n/a	58.62	58.17	58.95	60.15	60.61	59.62	59.57
WLK	63.18	64.60	64.33	65.37	63.09	63.33	63.07	62.59
WWLK	63.89	64.80	64.34	65.40	62.72	62.99	62.55	62.66
rematch	**64.93**	**65.88**	**65.06**	**66.52**	**67.03**	**67.72**	**67.10**	**67.34**

Table 2: Semantic consistency of AMR similarity metrics and AMR parsers on the test splits of STS-B and SICK-R datasets. Best results are highlighted in bold. *Sembleu* fails to parse some of the AMRs generated by *spring*.

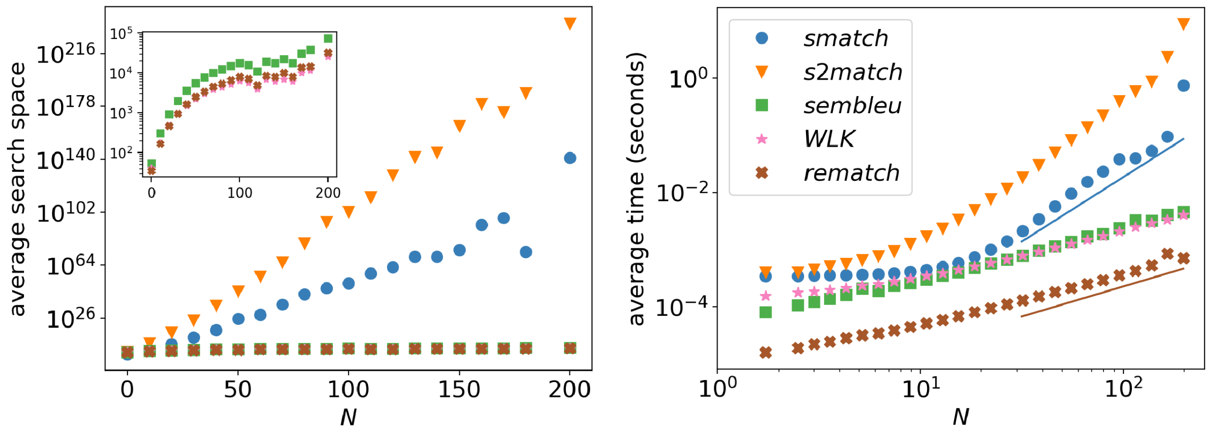

Figure 3: Average search space (left) and runtime (right) on a random sample of 500k pairs from AMR Annotation 3.0. N denotes the average size of each AMR pair. The inset zooms in on *sembleu*, *WLK*, and *rematch*, which cannot be distinguished in the log-linear plot. The lines on the runtime plot indicate approximate fits for $N > 10^{1.5}$, which on the log-log scale represent polynomial time complexity. The slopes indicate that the runtime scales quadratically for *smatch* ($O(N^{2.25})$) and linearly for *rematch* ($O(N)$).

Similarity Methods	STS-B	SICK-R
GloVe (avg.)	58.02	53.76
RoBERTa (first-last avg.)	58.55	61.63
AMR (*rematch*)	66.52	67.72
SimCSE-RoBERTa	**80.22**	**68.56**

Table 3: Comparison of similarity methods (AMR and non-AMR) on semantic consistency for the test splits of STS-B and SICK-R datasets.

WLK trailed further behind, requiring 927, 7718, and 315 seconds. All metrics executed the test bed on a single 2.25 GHz core. *Rematch*, *sembleu*, and *smatch* needed 0.2 GB of RAM, whereas *s2match* and *WLK* required 2 GB and 30 GB, respectively.

5.5 Ablation Study

To assess the impact of the three types of *rematch* motifs — attribute, instance, and relation — on structural and semantic similarity, let us conduct an ablation study, in which we remove one or more types of motifs at a time. The results are presented in Table 5. Instance motifs have the most significant influence on semantic similarity, particularly when combined with relation motifs. Conversely, relation motifs exert the strongest influence on structural similarity, especially when complemented by instance motifs.

To evaluate the overall effectiveness of motifs, we also assess the performance of *rematch* through the use of AMR labels alone. For the bottom AMR in Fig. 2, the label set is {`talk-01`, `person`, `politics`, `name`, *`ARG0`*, *`ARG1`*, *`ARG2`*, *`name`*, `-`, `"Helen"`, `"Maya"`, *`polarity`*, *`op1`*}. Note that `person`, `name`, and *`op1`* appear only once in the set. Similar to *rematch* motifs, we calculate the Jaccard similarity between two AMR label sets. As shown in Table 5, the decline in structural consistency when using AMR labels is substantial, given the absence of structural information in the label sets. In contrast, the decline in semantic consistency is relatively modest, indicating that AMR

	Main			Reification			Synonym Replace			Role Confusion			Avg.
	STS-B	SICK-R	PARA	STS-B	SICK-R	PARA	STS-B	SICK-R	PARA	STS-B	SICK-R	PARA	
smatch	53.01	57.65	40.96	53.02	59.74	40.08	51.76	55.42	39.60	**54.03**	**75.20**	24.78	50.44
s2match	55.87	57.38	**41.92**	55.41	59.46	**40.78**	54.86	56.12	**40.59**	48.23	73.89	**26.19**	**50.89**
sembleu	57.02	58.76	31.95	54.73	59.92	31.92	53.42	54.66	27.95	45.69	66.74	21.36	47.01
WLK	63.68	62.32	35.18	61.31	63.03	35.65	57.90	56.60	31.43	44.72	66.39	17.46	49.64
WWLK	**64.89**	62.92	37.72	61.90	**63.63**	37.39	**59.94**	57.47	33.35	5.50	43.55	0.09	44.03
rematch	64.72	**66.54**	34.88	**63.49**	62.55	35.82	59.75	**61.54**	32.70	42.38	67.28	15.37	50.59

Table 4: Hybrid consistency of AMR similarity metrics on the test split of the BAMBOO benchmark, for the three kinds of modifications, no modification (main) and the overall average. The best results are highlighted in bold.

	RARE	STS-B	SICK-R
rematch	95.01	73.95	71.01
– attribute	−00.85	−00.40	−00.09
– instance	+00.08	−06.34	−07.12
– relation	−62.30	+01.15	−01.41
– attribute, instance	+01.18	−16.78	−07.32
– attribute, relation	−62.55	+00.93	−01.92
– instance, relation	−95.87	−37.90	−62.32
labels	−72.89	−08.08	−07.30

Table 5: Ablation study of different motifs on structural (RARE) and semantic (STS-B, SICK-R) consistency. Dev splits of RARE and STS-B, and the trial split of SICK-R were used. The *mbse* and *amrbart* parsers were used for STS-B and SICK-R, respectively.

labels play a significant role in capturing semantics.

5.6 Error Analysis

Let us analyze a couple of examples where the semantic consistency of *rematch* significantly deviates from other AMR metrics. *Rematch* underestimates the semantic similarity between the sentences "Work into it slowly" and "You work on it slowly" compared to other metrics. This discrepancy arises from the AMR representation of the first sentence, which associates an attribute `imperative` with the verb `work-01`, a feature absent in the second sentence. Taking this attribute into account, *rematch* generates dissimilar instance motifs for the two AMRs.

In contrast, *rematch* accurately assigns a low similarity between the sentences "You should do it" and "You should never do it," unlike other metrics. It does so by capturing the negative (`-`) attribute `polarity`. Overall, we find that *rematch*'s scores are more conservative compared to other AMR metrics, particularly when dealing with negation.

6 Conclusion

In this paper, we introduce *rematch*, a novel and efficient metric for AMR similarity. *Rematch* leverages semantic AMR motifs to achieve superior performance compared to existing AMR metrics on both semantic consistency and computational efficiency. Furthermore, we introduce RARE, a new benchmark specifically designed to evaluate the structural consistency of AMR metrics. Using this benchmark, we find that *rematch* also exhibits strong sensitivity to structural transformations.

Despite its promising results, *rematch* has some limitations that merit future investigation. First, *rematch* only considers motifs associated with paths of length one (single edges). While this approach demonstrates strong performance on sentences, it might not capture higher-order semantics crucial for comparing AMRs derived from longer documents. Second, AMR-based similarity methods like *rematch* lag behind embeddings in capturing the semantic similarity of words. AMRs are limited in their ability to incorporate word-level similarity information, which is readily handled by embeddings. This can lead *rematch* to misclassify two sentences with different words as dissimilar even if their underlying concepts are equivalent.

Future research should delve deeper into the expressive power of Abstract Meaning Representations, particularly on their potential to address complex natural language understanding. Natural language inference (NLI) is one such task that is valuable for evaluating such AMR capabilities. Hybrid systems incorporating AMRs have already shown promise in this domain (Opitz et al., 2023). But a more intriguing avenue would be to explore methods that perform NLI entirely by exploiting the rich structure and semantics inherent in AMRs.

References

Eneko Agirre, Carmen Banea, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, Rada Mihalcea, German Rigau, and Janyce Wiebe. 2016. SemEval-2016 Task 1: Semantic Textual Similarity, Monolingual and Cross-Lingual Evaluation. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 497–511, San Diego, California. 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.

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. Number: 14.

Claire Bonial, Lucia Donatelli, Mitchell Abrams, Stephanie M. Lukin, Stephen Tratz, Matthew Marge, Ron Artstein, David Traum, and Clare Voss. 2020. Dialogue-AMR: Abstract Meaning Representation for Dialogue. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 684–695, Marseille, France. European Language Resources Association.

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.

Andrea Di Fabio, Simone Conia, and Roberto Navigli. 2019. VerbAtlas: a Novel Large-Scale Verbal Semantic Resource and Its Application to Semantic Role Labeling. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 627–637, Hong Kong, China. Association for Computational Linguistics.

William B. Dolan and Chris Brockett. 2005. Automatically Constructing a Corpus of Sentential Paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*.

Andrew Drozdov, Jiawei Zhou, Radu Florian, Andrew McCallum, Tahira Naseem, Yoon Kim, and Ramon Fernandez Astudillo. 2022. Inducing and Using Alignments for Transition-based AMR Parsing. ArXiv:2205.01464 [cs].

Angela Fan, Claire Gardent, Chloe Braud, and Antoine Bordes. 2019. Using Local Knowledge Graph Construction to Scale Seq2Seq Models to Multi-Document Inputs. *arXiv:1910.08435 [cs]*. ArXiv: 1910.08435.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2022. SimCSE: Simple Contrastive Learning of Sentence Embeddings. ArXiv:2104.08821 [cs].

Michael Wayne Goodman. 2020. Penman: An Open-Source Library and Tool for AMR Graphs. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 312–319, Online. 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.

Shu-Ming Hsieh, Chiun-Chieh Hsu, and Li-Fu Hsu. 2006. Efficient Method to Perform Isomorphism Testing of Labeled Graphs. In *Computational Science and Its Applications - ICCSA 2006*, Lecture Notes in Computer Science, pages 422–431, Berlin, Heidelberg. Springer.

Knight, Kevin, Badarau, Bianca, Baranescu, Laura, Bonial, Claire, Griffitt, Kira, Hermjakob, Ulf, Marcu, Daniel, O'Gorman, Tim, Palmer, Martha, Schneider, Nathan, and Bardocz, Madalina. 2020. Abstract Meaning Representation (AMR) Annotation Release 3.0. LDC Catalog No. LDC2020T02.

Young-Suk Lee, Ramon Fernandez Astudillo, Thanh Lam Hoang, Tahira Naseem, Radu Florian, and Salim Roukos. 2022. Maximum Bayes Smatch Ensemble Distillation for AMR Parsing. ArXiv:2112.07790 [cs].

Wai Ching Leung, Shira Wein, and Nathan Schneider. 2022. Semantic Similarity as a Window into Vector- and Graph-Based Metrics. In *Proceedings of the 2nd Workshop on Natural Language Generation, Evaluation, and Metrics (GEM)*, pages 106–115, Abu Dhabi, United Arab Emirates (Hybrid). Association for Computational Linguistics.

Fei Liu, Jeffrey Flanigan, Sam Thomson, Norman Sadeh, and Noah A. Smith. 2015. Toward Abstractive Summarization Using Semantic Representations. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1077–1086, Denver, Colorado. Association for Computational Linguistics.

Marco Marelli, Stefano Menini, Marco Baroni, Luisa Bentivogli, Raffaella Bernardi, and Roberto Zamparelli. 2014. A SICK cure for the evaluation of compositional distributional semantic models. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 216–223, Reykjavik, Iceland. European Language Resources Association (ELRA).

Arindam Mitra and Chitta Baral. 2016. Addressing a Question Answering Challenge by Combining Statistical Methods with Inductive Rule Learning and Reasoning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 30(1). Number: 1.

Tahira Naseem, Austin Blodgett, Sadhana Kumaravel, Tim O'Gorman, Young-Suk Lee, Jeffrey Flanigan, Ramón Astudillo, Radu Florian, Salim Roukos, and Nathan Schneider. 2022. DocAMR: Multi-sentence AMR representation and evaluation. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3496–3505, Seattle, United States. Association for Computational Linguistics.

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, Letitia Parcalabescu, and Anette Frank. 2020. AMR Similarity Metrics from Principles. *Transactions of the Association for Computational Linguistics*, 8:522–538. Place: Cambridge, MA Publisher: MIT Press.

Juri Opitz, Shira Wein, Julius Steen, Anette Frank, and Nathan Schneider. 2023. AMR4NLI: Interpretable and robust NLI measures from semantic graphs. ArXiv:2306.00936 [cs].

Martha Palmer, Daniel Gildea, and Paul Kingsbury. 2005. The Proposition Bank: An Annotated Corpus of Semantic Roles. *Computational Linguistics*, 31(1):71–106.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pages 311–318, USA. Association for Computational Linguistics.

Yisu Peng, Yuxiang Jiang, and Predrag Radivojac. 2018. Enumerating consistent sub-graphs of directed acyclic graphs: an insight into biomedical ontologies. *Bioinformatics*, 34(13):i313–i322.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Leonardo F. R. Ribeiro, Mengwen Liu, Iryna Gurevych, Markus Dreyer, and Mohit Bansal. 2022. FactGraph: Evaluating Factuality in Summarization with Semantic Graph Representations. ArXiv:2204.06508 [cs].

Nino Shervashidze, Pascal Schweitzer, Erik Jan van Leeuwen, Kurt Mehlhorn, and Karsten M. Borgwardt. 2011. Weisfeiler-Lehman Graph Kernels. *The Journal of Machine Learning Research*, 12(null):2539–2561.

Linfeng Song and Daniel Gildea. 2019. SemBleu: A Robust Metric for AMR Parsing Evaluation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4547–4552, Florence, Italy. Association for Computational Linguistics.

Nikhita Vedula and Srinivasan Parthasarathy. 2021. FACE-KEG: Fact Checking Explained using KnowledgE Graphs. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*, WSDM '21, pages 526–534, New York, NY, USA. Association for Computing Machinery.

Michihiro Yasunaga, Hongyu Ren, Antoine Bosselut, Percy Liang, and Jure Leskovec. 2021. QA-GNN: Reasoning with Language Models and Knowledge Graphs for Question Answering. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 535–546, Online. Association for Computational Linguistics.