PAM: Paraphrase AMR-Centric Evaluation Metric

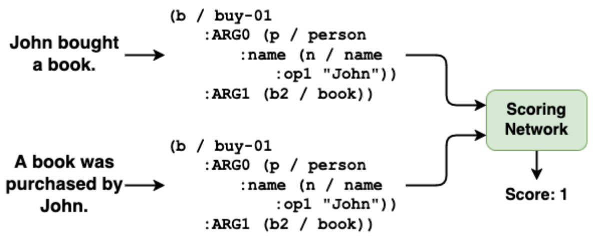
Afonso Sousa and **Henrique Lopes Cardoso**
Laboratório de Inteligência Artificial e Ciência de Computadores (LIACC)
Faculdade de Engenharia, Universidade do Porto, Portugal
{ammlss, hlc}@fe.up.pt

Abstract

Paraphrasing is rooted in semantics, which makes evaluating paraphrase generation systems hard. Current paraphrase generators are typically evaluated using borrowed metrics from adjacent text-to-text tasks, like machine translation or text summarization. These metrics tend to have ties to the surface form of the reference text. This is not ideal for paraphrases as we typically want variation in the lexicon while persisting semantics. To address this problem, and inspired by learned similarity evaluation on plain text, we propose PAM, a **P**araphrase **A**MR-Centric Evaluation **M**etric. This metric uses Abstract Meaning Representation (AMR) graphs extracted from the input text, which consist of semantic structures agnostic to the text surface form, making the resulting evaluation metric more robust to variations in syntax or lexicon. Additionally, we evaluated PAM on different semantic textual similarity datasets and found that it improves the correlations with human semantic scores when compared to other AMR-based metrics.

1 Introduction

Paraphrase generation is a relevant task in natural language processing (NLP), which has been widely applied in versatile tasks, such as question answering (Dong et al., 2017; Lan and Xu, 2018; Gan and Ng, 2019; Abujabal et al., 2019), machine translation (Madnani et al., 2012; Apidianaki et al., 2018; Kajiwara, 2019), and semantic parsing (Herzig and Berant, 2019; Wu et al., 2021; Cao et al., 2020). Recent years have witnessed rapid development in paraphrase generation algorithms (Bandel et al., 2022; Huang et al., 2022, 2023). However, little progress has been made in the automatic evaluation of this task. It is even unclear which metric is more reliable among many widely used metrics. Most evaluation metrics used in previous paraphrase generation research were not designed specifically for



```
(b / buy-01
   :ARG0 (p / person
      :name (n / name
         :op1 "John"))
   :ARG1 (b2 / book))
```

```
(b / buy-01
   :ARG0 (p / person
      :name (n / name
         :op1 "John"))
   :ARG1 (b2 / book))
```

Figure 1: Simplified outline of the approach with oracle example. The AMR parser would produce the same AMR graph for both paraphrases, and thus, the scoring is 1.

the task but adopted from other tasks, such as machine translation (MT) Papineni et al. (2002) and summarization (Lin, 2004). However, the evaluation of paraphrases is inherently different from the evaluation of most other text-generation tasks because a good paraphrase typically obeys two criteria (Gleitman, 1970; Chen and Dolan, 2011; Bhagat and Hovy, 2013): semantic similarity and lexical diversity. Semantic similarity means that the paraphrase maintains similar semantics to the input text, whereas lexical diversity requires that the paraphrase possesses lexical or syntactic differences from the input. In contrast, a task like machine translation does not have such lexical diversity requirements. It is, therefore, uncertain whether the metrics borrowed from other tasks are sensible in paraphrase evaluation. Recent semantic similarity metrics leverage pretrained language models to compute the graded similarities between two texts (Reimers and Gurevych, 2019). However, because of the way these models work, there will always be some attachment between the output score and the texts' specific ordering, structure, and wording. In hopes of reducing this coupling between semantics and syntax when learning a semantic similarity evaluation metric, we propose to explore *Abstract Meaning Representation* (AMR). Given that AMR is a formalism designed to capture the semantic content of a sentence, abstracting away from the surface form and focusing on the underlying meaning, we hypothesize that embedding representations of these structures instead of the traditional plain texts will help the trained models to better capture the semantics with less emphasis in the surface structure. We propose to learn text similarity through a similar pipeline used in plain-text automatic similarity assessment methods that encode sentences into latent semantic representations to measure the similarity of the two representations (Reimers and Gurevych, 2019; Gao et al., 2021); however, this time we encode latent semantic representations from AMR graphs extracted from the two input texts. Our proposed metric PAM, **P**araphrase **A**MR-Centric Evaluation **M**etric, builds on top of Sentence-BERT (Reimers and Gurevych, 2019), adding extra layers to process AMR graphs and fuse these newly computed graph representations into the text-based backbone. Inspired by other similarity scoring works (Gao et al., 2021; Shou and Lin, 2023), we utilize self-supervised learning methods to overcome the high cost of collecting training data. We experiment with PAM on two popular Semantic Textual Similarity (STS) datasets and demonstrate that PAM achieves considerable improvements in correlation with human annotations. In further analysis, PAM retains the highest performance under various challenges compared to previous metrics. The code to reproduce the experiments is available at `https://github.com/afonso-sousa/pam`.

2 Related Work

Neural networks have seen significant progress in paraphrase generation, leading to systems capable of producing more natural and human-like outputs (Sun et al., 2021; Huang and Chang, 2021; Ding et al., 2021; Wahle et al., 2023; Luo et al., 2023). Alongside these developments, extensive research has focused on automatically extracting paraphrases to build benchmarks, supporting further advancements in the field (Ganitkevitch et al., 2013; Pavlick et al., 2015; Pavlick and Callison-Burch, 2016; Zhang et al., 2019b; Dou et al., 2022; Huang et al., 2023). Despite these advancements, relatively little attention has been given to evaluating the quality of generated paraphrases. Earlier work (Goyal and Durrett, 2020; Kumar et al., 2020; Sun et al., 2021) typically relied on metrics that measure surface-level overlap between generated outputs and reference paraphrases, such as BLEU (Papineni et al., 2002), ROUGE (Lin, 2004), and METEOR (Banerjee and Lavie, 2005). However, these metrics are limited in their ability to capture semantic equivalence and lexical diversity – two key aspects of paraphrasing. As a result, recent research has increasingly shifted toward compound metrics that explicitly balance semantic fidelity with lexical variation (Hosking et al., 2022; Sousa and Lopes Cardoso, 2024a; Sousa and Cardoso, 2025). A prominent example is iBLEU (Sun and Zhou, 2012), which combines BLEU and self-BLEU (Shu et al., 2019) to quantify both meaning preservation and novelty. In parallel, modern embedding-based approaches like SBERT (Reimers and Gurevych, 2019) leverage sentence embeddings to measure semantic similarity, have also been employed for evaluating paraphrases as they offer robustness to lexical variations in paraphrases. AlignScore (Zha et al., 2023) further advances the field with a fact-aware alignment model that evaluates factual consistency between the source and paraphrase, addressing critical challenges in meaning preservation. To our knowledge, Shen et al. (2022) are the only researchers to systematically assess the adequacy of existing paraphrase evaluation metrics. They identified key shortcomings and introduced ParaScore, a composition of metrics combining semantic similarity and lexical divergence. We focus on improving the semantic similarity component of paraphrase evaluation, which can function independently or within a framework like ParaScore.

3 PAM

We introduce PAM, an evaluation metric that computes graded similarity scores between sentences using AMR graph representations. For each sentence, an AMR graph is extracted, linearized, and fed into a Siamese network, which outputs a similarity score as the cosine similarity between the graph embeddings.

3.1 Extracting AMR Graphs

AMR is a linguistically grounded semantic formalism that represents the meaning of a sentence as a rooted graph, where nodes are *concepts* and edges are *semantic relations*. AMR abstracts away from surface text, aiming to produce a more language-neutral representation of meaning. We use a stateof-the-art AMR parser[1] to extract an AMR graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{R})$ with node set $\mathcal{V}$ and labeled edges $(u, r, v) \in \mathcal{E}$, where $u, v \in \mathcal{V}$ and $r \in \mathcal{R}$. Each $\mathcal{G}$ provides an explicit representation of the core concepts in a sentence (refer to Figure 1 for an example of a sentence and its corresponding AMR graph).

Graph Representation Following Beck et al. (2018), we convert each graph $\mathcal{G}$ into its bipartite version $\mathcal{G}_b = (\mathcal{V}_b, \mathcal{E}_b)$, replacing each labeled edge $(u, r, v) \in \mathcal{E}$ with two unlabeled edges $(u, r), (r, v) \in \mathcal{E}_b$, effectively converting the original graph into an unlabeled graph. Additionally, pretrained models typically use a vocabulary with subword tokens. To be able to match the features of concept nodes with the representations of those subword tokens, we follow the work of Ribeiro et al. (2021), transforming each $\mathcal{G}_b$ into a new subword token graph $\mathcal{G}_t = (\mathcal{V}_t, \mathcal{E}_t)$, where each subword token of a node $v_b \in \mathcal{V}_b$ becomes a node $v_t \in \mathcal{V}_t$. We convert each edge $(u_b, v_b) \in \mathcal{E}_b$ into a set of edges and connect every token of u_b to every token of v_b. Following we show an example of this process:

1. Original Graph Let the original graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be:

$$\mathcal{V} = \{A, B, C\}, \quad \mathcal{E} = \{(A, r_1, B), (B, r_2, C)\}.$$

$$\begin{tikzcd} A \arrow[r, "r_1"] & B \arrow[r, "r_2"] & C \end{tikzcd}$$

2. Bipartite Graph Convert $\mathcal{G}$ to its bipartite version $\mathcal{G}_b$:

$$\mathcal{V}_b = \{A, B, C, r_1, r_2\}$$

$$\mathcal{E}_b = \{(A, r_1), (r_1, B), (B, r_2), (r_2, C)\}.$$

$$\begin{tikzcd} A \arrow[r] & r_1 \arrow[r] & B \arrow[r] & r_2 \arrow[r] & C \end{tikzcd}$$

3. Subword Token Graph Assume tokenization of nodes:

$$A = \{A_1, A_2\}, \quad B = \{B_1\}, \quad C = \{C_1, C_2\}.$$

Relation nodes remain unchanged: r_1, r_2. The subword token graph $\mathcal{G}_t$ becomes:

$$\mathcal{V}_t = \{A_1, A_2, B_1, C_1, C_2, r_1, r_2\},$$

$$\begin{aligned} \mathcal{E}_t = & \{(A_1, r_1), (A_2, r_1), (r_1, B_1), \\ & (B_1, r_2), (r_2, C_1), (r_2, C_2)\}. \end{aligned}$$

$$\begin{tikzcd} A_1 \arrow[dr] & & & & C_1 \\ & r_1 \arrow[r] & B_1 \arrow[r] & r_2 \arrow[ur] \arrow[dr] & \\ A_2 \arrow[ur] & & & & C_2 \end{tikzcd}$$

3.2 Model

Figure 2 illustrates PAM, which employs structural adapters (Ribeiro et al., 2021) to repurpose the pretrained encoder to structured inputs, in this case, AMR graphs.

Text Encoder To take advantage of the semantic understanding already present in the pretrained language model, we retain its main structures, namely the encoder, and repurpose it through fine-tuning. Instead of plain text, we feed it the linearized version of the AMR graph. These AMR graphs are linearized by a depth-first traversal algorithm. Additionally, while position embeddings are crucial for modeling sequence order in transformers, we argue that the linearization order should not affect AMR graph encoding. Due to the nature of AMR graphs, relationships between nodes are primarily defined by their semantic connections rather than their linear arrangement. Therefore, we remove the positional embeddings and keep just the regular subword embeddings. In transformer-based models like BERT, tokens may be split into smaller subwords due to the fixed vocabulary. Concept nodes are tokenized as usual. We do not split AMR relations and instead add them to the vocabulary as new words (Ribeiro et al., 2021; Shou and Lin, 2023). The main reason is that relation words are artificial, while nodes are always concepts, closer to linguistic tokens in the vocabulary.

Graph Encoder To embed explicitly the connectivity of the graphs in the model, PAM uses Graph Neural Networks (GNN). Specifically, it uses an adapter module similar to the work of Ribeiro et al. (2021), differing in having a single GNN layer per module, but stacking these modules, following Shou and Lin (2023) (see in Figure 2 the orange boxes. Each have a single GNN layer, but two are stacked together). The adapter modules are stacked and attached to the first and last pretrained encoder layers just before the feed-forward sub-layers (see in Figure 2 the green boxes. One replaces the very first encoder layer – blue boxes – and the other replaces the last encoder layer). The representation fed to these feed-forward sub-layers comes from a cross-attention layer that attends to both the text

[1]We use a BART-based (Lewis et al., 2020) model, the best-performing model from: `https://github.com/bjascob/amrlib`.

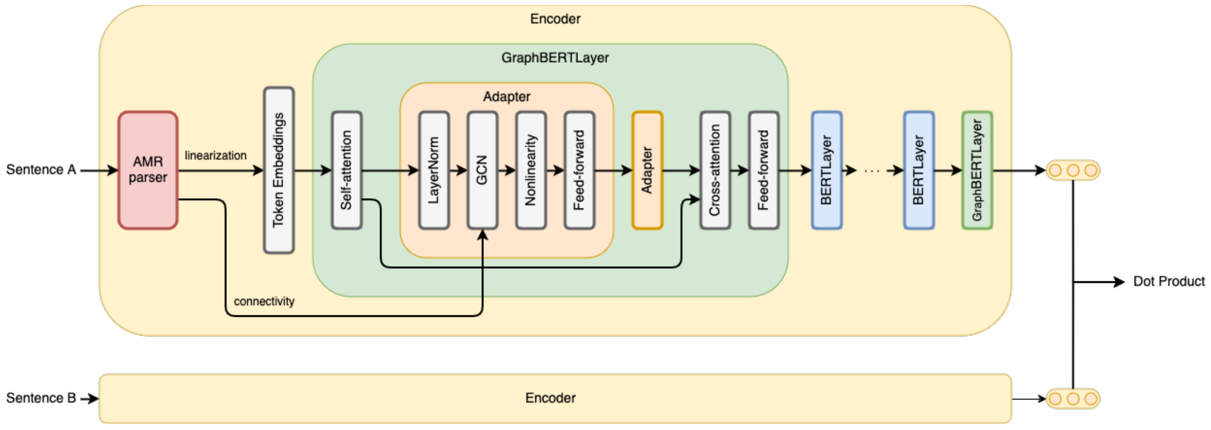


Figure 2: PAM architecture. The two encoder networks have tied weights (siamese network structure). The training objective is to maximize dot products between contextual embeddings from positive instances and minimize dot products between contextual embeddings from negative instances.

and graph representations. The graph features are populated with the hidden representations from the pretrained encoder at the respective encoder layer (notice in Figure 2 the information that goes into the GNN is the normalized self-attention from that layer). Specifically, for each node $v \in V$, given the hidden representation $\mathbf{h}_v^l$, the encoder layer l computes:

$$\mathbf{g}_v^l = \mathrm{GNN}_l\left(\mathrm{LN}(\mathbf{h}_v^l), \left\{\mathrm{LN}(\mathbf{h}_u^l) : u \in \mathcal{N}(v)\right\}\right),$$

$$\mathbf{z}_v^l = \mathbf{W}_e^l \sigma(\mathbf{g}_v^l) + \mathbf{h}_v^l,$$

where $\mathcal{N}(v)$ is the neighborhood of the node v in G and $\mathbf{W}_e^l \in \mathbb{R}^{d \times m}$ is an adapter parameter. $\mathrm{GNN}_l(\cdot)$ is the graph convolution that computes the representation of v based on its neighbors in the graph, and l is the index of the last encoder layer. We employ a Graph Convolutional Network (Kipf and Welling, 2016) as the graph neural network.

The final representation is computed with a pooling layer as in Reimers and Gurevych (2019).

3.3 Training Scheme

PAM is first pretrained using *self-supervision*, and then finetuned using *supervised learning*.

Self-supervised Training In plain text applications (e.g., STS and text generation tasks), many learned metrics are trained to optimize correlation with human annotations (Reimers and Gurevych, 2019). However, data collection on AMR graph similarity is more time-consuming because AMR evaluation has a learning cost of understanding the semantics of graphs, which are not as straightforward as plain text (Shou and Lin, 2023). Thus, self-supervised learning methods are an alternative solution. We adopt Contrastive Tension (CT) (Carlsson et al., 2021). CT constructs positive and negative pairs as follows: Each randomly selected AMR graph $\mathcal{G}$ is paired with itself to construct a positive instance; other $\mathcal{K}$ graphs are sampled to construct negative instances by pairing them with $\mathcal{G}$. The assumption for generating negative samples is that two randomly selected sentences are likely to be semantically dissimilar. The $\mathcal{K}+1$ instances are included in the same batch. The training contrastive loss $\mathcal{L}$ is the binary cross-entropy applied to the generated similarity scores, taking into account the positive/negative labels. The training objective using CT is to maximize the dot product between positive instance representations and to minimize the dot product between negative instance representations. The loss function is thus given by:

$$L(\mathcal{G}, \hat{\mathcal{G}}) = \begin{cases} -\log \sigma(\mathbf{e} \cdot \hat{\mathbf{e}}), & \text{if } \mathcal{G} = \hat{\mathcal{G}} \\ -\log \sigma(1 - \mathbf{e} \cdot \hat{\mathbf{e}}), & \text{if } \mathcal{G} \neq \hat{\mathcal{G}} \end{cases}$$

where σ is the Logistic function. A pair of AMR graphs is fed into the siamese network, one for each branch, to generate their respective contextual embeddings, from which we compute the dot product and loss (refer to Figure 2 for an illustration).

Supervised Learning with Triplets In addition to self-supervised learning, we incorporate supervised learning via a triplet-based learning approach (Reimers and Gurevych, 2019), where the training objective is to optimize the relative similarity relationships among an anchor ($\mathcal{A}$, a reference AMR graph), a positive ($\mathcal{P}$, an AMR graph semantically similar to the anchor), and a negative ($\mathcal{N}$, an AMR graph semantically dissimilar to the anchor). The triplet loss encourages the anchor-positive embedding similarity to increase while reducing similarity with the negative:

$$\mathcal{L}_{\text{triplet}} = \max\big(0, m + d(\mathbf{e}_{\mathcal{A}}, \mathbf{e}_{\mathcal{P}}) - d(\mathbf{e}_{\mathcal{A}}, \mathbf{e}_{\mathcal{N}})\big),$$

where $d(\mathbf{e}_i, \mathbf{e}_j)$ denotes the distance between embeddings $\mathbf{e}_i$ and $\mathbf{e}_j$ (e.g., cosine or Euclidean distance), and m is the margin hyperparameter that defines the minimum required distance difference between positive and negative pairs. The training batches were built utilizing an online mining strategy that dynamically selects "hard" examples. Hard positives are anchor-positive pairs $(\mathcal{A}, \mathcal{P})$ that are difficult to distinguish due to high embedding similarity, while hard negatives are anchor-negative pairs $(\mathcal{A}, \mathcal{N})$ that are challenging due to low embedding dissimilarity. For each triplet, the anchor, positive, and negative AMR graphs are processed through the branches of the Siamese network to generate contextual embeddings. These embeddings are then used to compute the triplet loss.

4 Training Data

To train PAM, we first use data parsed by Shou and Lin (2023), which entails one million sentences (randomly sampled from English Wikipedia, used in SimCSE (Gao et al., 2021)) parsed to AMR graphs using SPRING (Bevilacqua et al., 2021). This data is used in the self-supervised training, where we set the positive ratio to be 4/16, following Shou and Lin (2023), in a batch of 16, there are 4 positive graph pairs and 12 negative pairs. That is, in a batch we sample 4 graphs and create one positive pair and three negative pairs for each graph. We further train PAM in a supervised learning fashion using 101 762 entries from the Quora Question Pairs (QQP) dataset[2]. These entries were compiled by Reimers and Gurevych (2019) as anchor-positive-negative triplets[3].

5 Semantic Textual Similarity Results

Following Shen et al. (2022), we select, as baselines, well-known metrics often used to evaluate paraphrase generation. We categorize evaluation metrics into three main types based on their focus:

Lexical Overlap: BLEU (Papineni et al., 2002) (n-gram precision with brevity penalty), ROUGE (Lin, 2004) (e.g., ROUGE-L, measuring recall of overlapping n-grams), and METEOR (Banerjee and Lavie, 2005) (precision, recall, and synonym matching).

Token-Level Semantic Similarity: BERTScore (Zhang et al., 2019a) (contextual token similarity using BERT) and BARTScore (Yuan et al., 2021) (text quality scoring with BART).

Semantic Structure Embedding: SBERT (Reimers and Gurevych, 2019) (sentence embeddings for semantic similarity) and AlignScore (Zha et al., 2023) (cross-task alignment modeling for factual verification).

We compare PAM with the aforementioned metrics on modified test sets from two popular semantic textual similarity datasets, STSb (Cer et al., 2017) and SICK (Marelli et al., 2014). The original datasets contain pairs of sentences with human-labeled similarity scores. Opitz et al. (2021) utilized a parser to construct AMR graph pairs from those sentence pairs and further confirmed the overall quality of those generations. To evaluate PAM for STS, we skip the AMR parsing from the input sentence and use these AMR graphs directly. These datasets have, respectively, 1 379 and 4 927 test instances. Table 1 shows the Pearson correlation between generations and references for the various metrics on the two test datasets. PAM outperforms all other metrics by a significant margin, achieving the highest score on correlation with human annotation. Specifically, PAM improves over SBERT from 65.55% to 75.73% on STSb, and from 71.82% to 73.43% on the SICK dataset.

6 Reframing Robustness Analysis

A basic assumption of PAM is that it is concerned with the entire AMR graph and is unaffected by *focus* change – or reframing – of the AMR graph, that is, restructuring the graph by having as root node a different concept. For this, we create synthetic versions of the STSb and SICK datasets, where, for each entry pair, we compute all possible rearrangements of the AMR graphs by changing its *focus* (Huang et al., 2023) to other concepts (see reframing examples in Figure 3). We randomly sample these reframed versions of the original AMR and pair them randomly, creating up to three new entry pairs per original pair. Since PAM can also be

[2]`https://quoradata.quora.com/First-Quora-Dataset-Release-Question-Pairs`
[3]`https://huggingface.co/datasets/sentence-transformers/quora-duplicates`

Metric	STSb	SICK
lexical overlap		
BLEU	50.58	52.53
ROUGE-L	50.92	56.72
METEOR	46.48	54.24
token-level semantic similarity		
BERTScore	33.15	45.22
BARTScore	51.23	58.47
semantic structure embedding		
AlignScore	41.64	50.82
SBERT	65.55	71.82
PAM	**75.73**	**73.43**

Table 1: Pearson correlation ρ between generated and reference scores on STSb and SICK test sets. Performance is reported by convention as $\rho * 100$. The best results are in **bold**.

seen as an AMR similarity metric, we assess its performance against various metrics often used to evaluate AMR similarity:

SMATCH (Cai and Knight, 2013): This metric evaluates the structural overlap between AMR graphs by treating each graph as a conjunction of triples. SMATCH identifies a one-to-one variable matching that maximizes the number of exact triple matches using a greedy hill-climbing algorithm.

SemBLEU (Song and Gildea, 2019): This metric extends BLEU (Papineni et al., 2002) to AMR graph comparison. SemBLEU linearizes AMR graphs through breadth-first traversal and assesses text quality by comparing n-grams.

WWLK (Opitz et al., 2021): Weisfeiler-Leman AMR similarity metric view AMR graphs as high-dimensional objects. The method iteratively propagates node embeddings with contextualization and employs the Wasserstein Weisfeiler-Leman kernel (WWLK) to compute the minimum cost of transforming one graph into another.

Table 2 shows the results for these baseline metrics for STSb and SICK test sets. The table includes AMRSim (Shou and Lin, 2023), a work similar to PAM that repurposes the SBERT architecture for AMR similarity evaluation. For the reframing dataset, we train and test an AMRSim model following the original paper setup. PAM achieves the highest scores across datasets and variants tested.

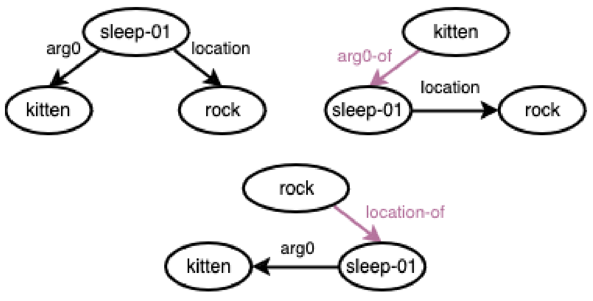


Figure 3: Example of reframing transformations of the same AMR graph.

Specifically for the reframing dataset, since it has some structures that were never seen during training, a decrease in performance is expected. Notably, when comparing PAM with AMRSim, the performance drop (Δ) is -1.63 vs. -4.12 on STSb and -1.97 vs. -2.75 on SICK, highlighting PAM's superior robustness.

Metric	Original		Reframing	
	STSb	SICK	STSb	SICK
SMATCH	57.82	60.41	64.87	59.19
SemBLEU	38.07	35.32	40.63	36.16
WWLK	50.41	50.92	54.32	53.95
AMRSim	70.88	73.10	66.76	70.35
PAM	**75.73**	**73.43**	**73.90**	**71.46**

Table 2: Performance comparison of PAM against AMR similarity metrics for original and reframed datasets. Results are Pearson correlation (x100) on STSb and SICK test sets. The best results are in **bold**.

7 Ablation Study

Model	STSb (Δ)	SICK (Δ)
PAM w/o supervised learning	72.59	72.23
w/o cross-attention	70.06 (**-2.53**)	71.54 (**-0.69**)
w/o graph adapter	67.95 (**-4.64**)	70.25 (**-1.98**)

Table 3: Ablation study. Results are reported as Person correlation (x100) and differences (Δ) to the full model (PAM) on STSb and SICK test datasets.

To understand the contribution of each component of the approach, we conduct an ablation study whose results are shown in Table 3. We evaluate PAM without supervised learning to reduce computational costs. All components contribute positively to final performance.

Without Cross-Attention: This variant removes

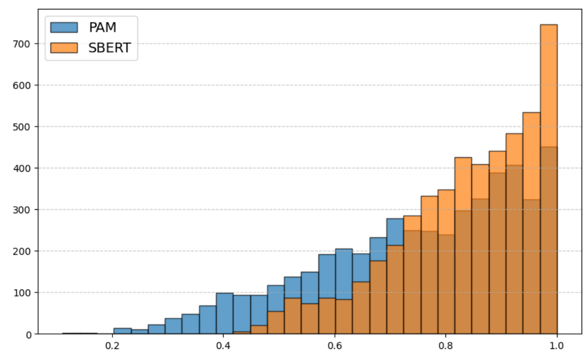


Figure 4: Distribution of scores of PAM and SBERT for the SICK test set.

the cross-attention mechanism, fusing the linearized graph textual representation and graph representation with concatenation. It performs worse than the full model, showing a score drop of 2.53 on STSb and 0.69 on SICK, highlighting the importance of cross-attention in integrating graph information effectively.

Without Graph Adapter: This variant removes all the additional parameters introduced by PAM, reverting the model to a plain SBERT configuration where the textual input consists of linearized AMR graphs. The performance of this variant degrades significantly compared to the full model, with a score drop of 4.64 on the STSb dataset and 1.98 on the SICK dataset. This demonstrates that the explicit connectivity embedded into the model through GNNs is crucial for its performance.

8 Score Distribution

We compare the score distributions of PAM and SBERT. As observed in the histogram in Figure 4, PAM shows a higher dispersion of scores when compared with SBERT. PAM's greater dispersion provides finer granularity for distinguishing paraphrase quality, enabling it to more easily capture subtle differences between near-paraphrases, partial paraphrases, and unrelated pairs. In contrast, SBERT's concentrated distribution risks overstating similarity for non-paraphrastic pairs and may fail to penalize meaningful deviations.

9 Performance over Paraphrase Types

9.1 In-Depth Analysis of Paraphrastic Phenomena

We evaluate PAM's paraphrase assessment using examples from the seminal work of Bhagat and Hovy (2013), covering various paraphrase types. alongside their corresponding semantic similarity scores. These examples consist of sentence or phrase pairs that convey approximately the same meaning using different words, though they may not qualify as strict paraphrases. To complement these examples, we manually construct non-paraphrases by altering their meaning with minimal changes. Table 4 shows the scores assigned by PAM and SBERT for the various examples. Following, we mainly discuss the scores for the paraphrase column, as the non-paraphrase is less nuanced.

Synonym substitution: While *bought* and *acquired* convey a similar meaning in this context, implying a transfer of ownership, their exact meanings have subtle distinctions. This nuance is reflected in the paraphrase evaluation score of 94.16 compared to 98.99, where a slightly lower score for this substitution is arguably more appropriate. Their vector representations in static word embeddings show a cosine similarity of 0.3980, indicating moderate semantic similarity. This supports the slight penalty, as the words are related but not perfectly interchangeable.

Converse substitution: The sentences describe the same event, differing only in grammatical structure (active vs. passive voice). This makes them near-perfect paraphrases. The score of 90.53, while high, may underrepresent the strong semantic equivalence between the two sentences. In contrast, the score of 95.46 better reflects their similarity, as the grammatical shift does not alter their meaning in this context.

Change of voice: This example illustrates the oracle example in Figure 1, as the two sentences produce the same AMR graph:

```
(l / love-01
  :ARG0 (p / person
        :name (n / name
              :op1 "Pat"))
  :ARG1 (p2 / person
        :name (n2 / name
              :op1 "Chris")))
```

A perfect similarity score is thus expected as the grammatical shift does not alter the sentence meaning in this context.

Pronoun/Co-referent substitution: Yet again, here, the grammatical shift does not alter the sentence meaning in this context. Thus, a perfect score aligns with the expected outcome. This perfect score comes from the same AMR representing both

Phenomenon	Paraphrase (PAM/SBERT)	Non-Paraphrase (PAM/SBERT)
Synonym substitution	Google bought YouTube. ⇔ Google acquired YouTube. (**94.16**/98.99)	Google bought YouTube. ⇎ Google destroyed YouTube. (86.30/**82.50**)
Converse substitution	Google bought YouTube. ⇔ YouTube was sold to Google. (90.53/**95.46**)	Google bought YouTube. ⇎ Google rented YouTube. (**81.45**/90.64)
Change of voice	Pat loves Chris. ⇔ Chris is loved by Pat. (**100.0**/96.76)	Pat loves Chris. ⇎ Chris hates Pat. (**60.74**/78.28)
Pronoun/Co-referent substitution	Pat likes Chris, because she is smart. ⇔ Pat likes Chris, because Chris is smart (**100.0**/96.21)	Pat likes Chris, because she is smart. ⇎ Pat likes Chris, because Pat is smart (99.97/**96.18**)
Actor/Action substitution	I dislike rash drivers. ⇔ I dislike rash driving (**87.77**/97.76)	I dislike rash drivers. ⇎ I dislike drivers who follow rules (**82.85**/87.35)

Table 4: Paraphrase phenomena examples with paraphrase and non-paraphrase counterparts, including similarity scores. *Paraphrase* pairs are retrieved from the work of Bhagat and Hovy (2013). We highlight the scores we deem best in **bold**.

sentences:

```
(l / like-01
   :ARG0 (p / person
         :name (n / name
               :op1 "Pat"))
   :ARG1 (p2 / person
         :name (n2 / name
               :op1 "Chris"))
   :ARG1-of (c / cause-01
         :ARG0 (s / smart-06
               :ARG1 p2)))
```

In the negative example, the only change in the AMR is `:ARG1 p2` ⇒ `:ARG1 p`, which the model does not understand as a big change and gives it a high score.

Actor/Action substitution: The sentences share strong semantic overlap but differ in focus: one emphasizes individuals, the other behavior. Although related, this shift introduces a subtle distinction. The score of 87.77 accurately reflects their close but not identical meanings, whereas 97.76 overstates their similarity by ignoring this nuance.

From the abovementioned discussion, our main takeaway is that PAM excels at capturing fine-grained distinctions (e.g., synonym substitution or actor/action substitution) and strict paraphrases (e.g., change of voice or pronoun/co-referent substitution), which makes it a suitable metric for paraphrase generation evaluation. SBERT may be more suited for coarse-grained paraphrase identification where near interchangeability might be more desirable (like ignoring the nuanced distinction between "bought" and "acquired" in synonym substitution).

9.2 Automatic Evaluation on Paraphrase Types

To ensure that our conclusions extend beyond individual examples, we compare PAM and SBERT on the Extended Paraphrase Typology Corpus (ETPC) (Kovatchev et al., 2018), a dataset of parallel paraphrases annotated with 26 fine-grained paraphrase types (including non-paraphrases).

Figure 5 shows the normalized (Z-score) scores for PAM and SBERT on the different types. Consistent with our earlier findings (§9.1), PAM assigns lower scores to *converse substitutions*. Similarly, PAM penalizes *opposite polarity substitutions* and *semantic-based* paraphrases more than SBERT. Interestingly, all these types are labeled in the original dataset paper (Kovatchev et al., 2018, Table 1) as cases that may produce both paraphrases and non-paraphrases, which may justify the differing view of both metrics. A type linked to non-paraphrases, as per the authors of the dataset, is *entailment*. Here, PAM correctly assigns lower similarity scores, whereas SBERT overestimates similarity. *Non-paraphrases* are also better evaluated by PAM. For the types associated with textual paraphrases that got the highest score difference between the two metrics, *ellipsis* and *coordination changes*, PAM provides more appropriate scores.

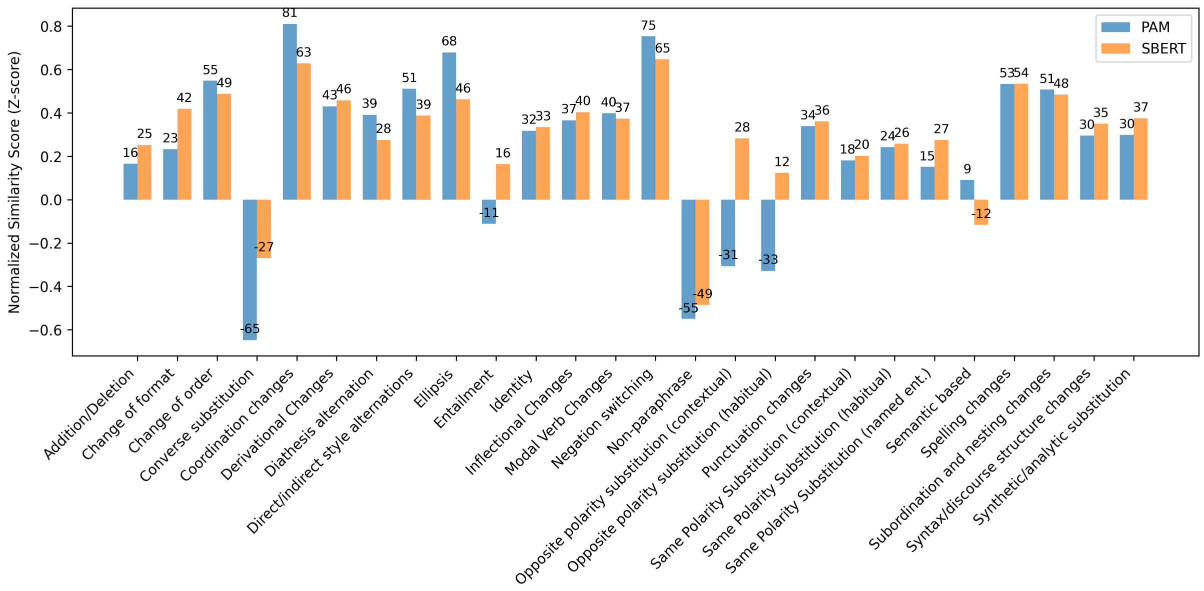


Figure 5: Normalized distribution of scores of PAM and SBERT for the ETPC dataset per paraphrase type.

PAM as a Binary Classifier To further evaluate PAM 's effectiveness, we tested its performance as a binary classifier. By selecting a decision threshold that corresponds to a fixed false positive rate (FPR) of 5%, we assessed its true positive rate (TPR) and precision, comparing it directly to SBERT. This approach allows us to analyze how well each method balances sensitivity and specificity, providing insights into their practical utility in scenarios where controlling false positives is critical. PAM achieves a TPR of 0.321 and precision of 0.929, outperforming SBERT, which achieves a TPR of 0.243 and precision of 0.908. These findings suggest that, at an equivalent false positive rate, PAM is more effective at correctly identifying positive instances (higher TPR) while also maintaining a slightly higher precision.

10 Conclusion

We have proposed a learning-based semantic similarity metric for paraphrase evaluation called PAM. It leverages AMR graphs that are extracted from input texts by a pretrained AMR parser and processed by the model's underlying siamese network structure that incorporates GNNs for explicit graph connectivity representation. PAM is trained under self-supervised and supervised learning. Our proposed similarity metric addresses the main problem with currently employed metrics for paraphrase generation evaluation: the assumption of a high coupling between semantic meaning and surface form. PAM effectively encodes the meaning of the texts by leveraging AMR graphs, attaining the highest correlations with human evaluations compared to other test metrics, and maintaining robustness against reframed AMR variants. It also has a more discriminative distribution than SBERT.

Limitations

Throughout this document, we have already acknowledged some of the limitations of PAM. The main limitation of this work is that it heavily relies on the pretrained AMR parser. The improper extraction of AMR graphs can affect the overall performance of the model as it fails to produce accurate representations. The required extraction of AMR graphs also entails a substantial computational overhead that might make the metric unfit for some use cases. Finally, due to its reliance on the AMR parser, our approach may not be easily employed for specialized domains or languages other than English, for which appropriate AMR parsers may not be available.

Acknowledgments

The first author is supported by a PhD studentship with reference 2022.13409.BD from Fundação para a Ciência e a Tecnologia (FCT). This work was financially supported by: UID/00027 of the LIACC – Artificial Intelligence and Computer Science Laboratory – funded by Fundação para a Ciência e a Tecnologia, I.P./ MCTES through the national funds.

References

Abdalghani Abujabal, Rishiraj Saha Roy, Mohamed Yahya, and Gerhard Weikum. 2019. Comqa: A community-sourced dataset for complex factoid question answering with paraphrase clusters. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 307–317.

Marianna Apidianaki, Guillaume Wisniewski, Anne Cocos, and Chris Callison-Burch. 2018. Automated paraphrase lattice creation for hyter machine translation evaluation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 480–485.

Elron Bandel, Ranit Aharonov, Michal Shmueli-Scheuer, Ilya Shnayderman, Noam Slonim, and Liat Ein-Dor. 2022. Quality controlled paraphrase generation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 596–609, Dublin, Ireland. Association for Computational Linguistics.

Satanjeev Banerjee and Alon Lavie. 2005. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, pages 65–72.

Daniel Beck, Gholamreza Haffari, and Trevor Cohn. 2018. Graph-to-sequence learning using gated graph neural networks. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 273–283, Melbourne, Australia. 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. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 12564–12573.

Rahul Bhagat and Eduard Hovy. 2013. Squibs: What is a paraphrase? *Computational Linguistics*, 39(3):463–472.

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.

Ruisheng Cao, Su Zhu, Chenyu Yang, Chen Liu, Rao Ma, Yanbin Zhao, Lu Chen, and Kai Yu. 2020. Unsupervised dual paraphrasing for two-stage semantic parsing. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6806–6817.

Fredrik Carlsson, Amaru Cuba Gyllensten, Evangelia Gogoulou, Erik Ylipää Hellqvist, and Magnus Sahlgren. 2021. Semantic re-tuning with contrastive tension. In *International Conference on Learning Representations*.

Daniel Cer, Mona Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. 2017. SemEval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 1–14, Vancouver, Canada. Association for Computational Linguistics.

David Chen and William B Dolan. 2011. Collecting highly parallel data for paraphrase evaluation. In *Proceedings of the 49th annual meeting of the association for computational linguistics: human language technologies*, pages 190–200.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Kaize Ding, Dingcheng Li, Alexander Hanbo Li, Xing Fan, Chenlei Guo, Yang Liu, and Huan Liu. 2021. Learning to selectively learn for weakly-supervised paraphrase generation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5930–5940, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Li Dong, Jonathan Mallinson, Siva Reddy, and Mirella Lapata. 2017. Learning to paraphrase for question answering. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 875–886.

Yao Dou, Chao Jiang, and Wei Xu. 2022. Improving large-scale paraphrase acquisition and generation. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 9301–9323, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Wee Chung Gan and Hwee Tou Ng. 2019. Improving the robustness of question answering systems to question paraphrasing. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6065–6075.

Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. 2013. Ppdb: The paraphrase database. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 758–764.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. SimCSE: Simple contrastive learning of sentence embeddings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6894–6910, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Lila R Gleitman. 1970. Phrase and paraphrase-some innovative uses of language. *WW Norten & Company Inc*.

Tanya Goyal and Greg Durrett. 2020. Neural syntactic preordering for controlled paraphrase generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 238–252.

Jonathan Herzig and Jonathan Berant. 2019. Don't paraphrase, detect! rapid and effective data collection for semantic parsing. 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 3810–3820.

Tom Hosking, Hao Tang, and Mirella Lapata. 2022. Hierarchical sketch induction for paraphrase generation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2489–2501, Dublin, Ireland. Association for Computational Linguistics.

Kuan-Hao Huang and Kai-Wei Chang. 2021. Generating syntactically controlled paraphrases without using annotated parallel pairs. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1022–1033.

Kuan-Hao Huang, Varun Iyer, I-Hung Hsu, Anoop Kumar, Kai-Wei Chang, and Aram Galstyan. 2023. ParaAMR: A large-scale syntactically diverse paraphrase dataset by AMR back-translation. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8047–8061, Toronto, Canada. Association for Computational Linguistics.

Kuan-Hao Huang, Varun Iyer, Anoop Kumar, Sriram Venkatapathy, Kai-Wei Chang, and Aram Galstyan. 2022. Unsupervised syntactically controlled paraphrase generation with Abstract Meaning Representations. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 1547–1554, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Tomoyuki Kajiwara. 2019. Negative lexically constrained decoding for paraphrase generation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6047–6052.

Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

Venelin Kovatchev, M. Antònia Martí, and Maria Salamó. 2018. ETPC - a paraphrase identification corpus annotated with extended paraphrase typology and negation. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Ashutosh Kumar, Kabir Ahuja, Raghuram Vadapalli, and Partha Talukdar. 2020. Syntax-guided controlled generation of paraphrases. *Transactions of the Association for Computational Linguistics*, 8:330–345.

Wuwei Lan and Wei Xu. 2018. Neural network models for paraphrase identification, semantic textual similarity, natural language inference, and question answering. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3890–3902.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Yinhan Liu. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 364.

Haotian Luo, Yixin Liu, Peidong Liu, and Xianggen Liu. 2023. Vector-quantized prompt learning for paraphrase generation. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 13389–13398, Singapore. Association for Computational Linguistics.

Nitin Madnani, Joel Tetreault, and Martin Chodorow. 2012. Re-examining machine translation metrics for paraphrase identification. In *Proceedings of the 2012 conference of the north american chapter of the association for computational linguistics: Human language technologies*, pages 182–190.

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).

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.

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 of the Association for Computational Linguistics*, pages 311–318.

Ellie Pavlick and Chris Callison-Burch. 2016. Simple ppdb: A paraphrase database for simplification. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 143–148.

Ellie Pavlick, Pushpendre Rastogi, Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. 2015. Ppdb 2.0: Better paraphrase ranking, fine-grained entailment relations, word embeddings, and style classification. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 425–430.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. 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 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Leonardo F. R. Ribeiro, Yue Zhang, and Iryna Gurevych. 2021. Structural adapters in pretrained language models for AMR-to-Text generation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4269–4282, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Lingfeng Shen, Lemao Liu, Haiyun Jiang, and Shuming Shi. 2022. On the evaluation metrics for paraphrase generation. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 3178–3190, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Ziyi Shou and Fangzhen Lin. 2023. Evaluate AMR graph similarity via self-supervised learning. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 16112–16123, Toronto, Canada. Association for Computational Linguistics.

Raphael Shu, Hideki Nakayama, and Kyunghyun Cho. 2019. Generating diverse translations with sentence codes. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1823–1827.

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.

Afonso Sousa and Henrique Cardoso. 2025. Sapg: Semantically-aware paraphrase generation with amr graphs. In *Proceedings of the 17th International Conference on Agents and Artificial Intelligence - Volume 2: ICAART*, pages 861–871. INSTICC, SciTePress.

Afonso Sousa and Henrique Lopes Cardoso. 2024a. Pseudo-semantic graphs for generating paraphrases. In *EPIA Conference on Artificial Intelligence*, pages 215–227. Springer.

Afonso Sousa and Henrique Lopes Cardoso. 2024b. Pseudo-semantic graphs for generating paraphrases. In *Progress in Artificial Intelligence: 23rd EPIA Conference on Artificial Intelligence, EPIA 2024, Viana Do Castelo, Portugal, September 3–6, 2024, Proceedings, Part III*, page 215–227, Berlin, Heidelberg. Springer-Verlag.

Hong Sun and Ming Zhou. 2012. Joint learning of a dual SMT system for paraphrase generation. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 38–42, Jeju Island, Korea. Association for Computational Linguistics.

Jiao Sun, Xuezhe Ma, and Nanyun Peng. 2021. Aesop: Paraphrase generation with adaptive syntactic control. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5176–5189.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph attention networks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*.

Jan Philip Wahle, Bela Gipp, and Terry Ruas. 2023. Paraphrase types for generation and detection. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 12148–12164, Singapore. Association for Computational Linguistics.

Shan Wu, Bo Chen, Chunlei Xin, Xianpei Han, Le Sun, Weipeng Zhang, Jiansong Chen, Fan Yang, and Xunliang Cai. 2021. From paraphrasing to semantic parsing: Unsupervised semantic parsing via synchronous semantic decoding. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5110–5121.

Weizhe Yuan, Graham Neubig, and Pengfei Liu. 2021. Bartscore: Evaluating generated text as text generation. *Advances in Neural Information Processing Systems*, 34.

Yuheng Zha, Yichi Yang, Ruichen Li, and Zhiting Hu. 2023. AlignScore: Evaluating factual consistency with a unified alignment function. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11328–11348, Toronto, Canada. Association for Computational Linguistics.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2019a. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*.

Yuan Zhang, Jason Baldridge, and Luheng He. 2019b. PAWS: Paraphrase adversaries from word scrambling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1298–1308, Minneapolis, Minnesota. Association for Computational Linguistics.

A Experimental Setup

We implemented PAM with sentence transformers (Reimers and Gurevych, 2019). The transformer parameters were initialized from the uncased BERT base model (Devlin et al., 2019)[4], and the remaining parameters were initialized randomly. We followed the setup from Shou and Lin (2023), and set an input length of linearized AMR graphs as 128, learning rate as 1e-5, dropout rate as 0.1, graph adapter size as 128, and trained for 1 epoch. The optimizer we used was AdamW. We stacked two adapter modules.

B Sensitivity Analysis

Model	STSb (Δ)	SICK (Δ)
PAM w/o supervised learning	72.59	72.23
Architecture tuning		
init with ROBERTA	9.74 (**-62.85**)	15.13 (**-57.1**)
GAT as GNN layer	69.89 (**-2.7**)	71.67 (**-0.92**)
adapter at every other layer	68.54 (**-4.05**)	70.14 (**-2.09**)
Hyperparameter tuning		
adapter stack size: 3	70.91 (**-1.68**)	71.19 (**-1.04**)
adapter stack size: 4	71.03 (**-1.56**)	72.07 (**-0.16**)
GNN hidden size: 256	69.84 (**-2.75**)	71.90 (**-0.33**)

Table 5: Sensitivity analysis. Results are reported as Person correlation (x100) and differences (Δ) to the full model (PAM) without the supervised learning step on STSb and SICK test datasets.

To understand the contribution of each component of the approach, we conduct a sensitivity analysis whose results are shown in Table 5:

1. **Initialized with RoBERTa**: This variant initializes the PLM layers with RoBERTa (Liu, 2019) instead of BERT. Interestingly, using RoBERTa with our training setup (see Appendix A) significantly reduces performance. We attribute this drastic drop to RoBERTa's increased sensitivity to hyperparameters. Further investigation into hyperparameter tuning may yield additional insights.

2. **GAT as GNN Layer**: This variant replaces the GCN layer with a Graph Attention Network (GAT) (Veličković et al., 2018). Besides introducing additional computational costs due to multi-head attention, this layer yields poorer results, with a drop of 2.7 on STSb and 0.92 on SICK. This suggests the limited usefulness of GAT in this context.

3. **Adapter at every other layer**: Adding the adapter layer stack to every other encoder layer results in a performance drop in STSb (4.05) and SICK (2.09). This suggests that the model loses performance under the used training settings if too many parameters are added. A longer training process might help these variants recover some of the lost performance.

4. **Adapter stack size of 3**: Increasing the number of adapter layers to 3 causes a performance drop of 1.68 on STSb and 1.04 on SICK. This suggests that adding weights may be leading to model overfitting.

5. **Adapter stack size of 4**: Increasing the number of adapter layers to 4 results in a performance reduction of 1.56 on STSb and 0.16 on SICK. This suggests that adding weights may be leading to model overfitting.

6. **GNN hidden size of 256**: Increasing the hidden size of the GNN layer to 256 leads to a score reduction of 2.75 on STSb and 0.33 on SICK. Similar to the aforementioned variants, this demonstrates the sensitivity of the model to increasing parameters, suggesting it is leading the model to forget pre-acquired knowledge from the PLM layers.

The results in Table 5 are comparing performance with PAM without supervised learning. This choice was solely based on reducing the expense of training more models and does not deter from the findings. The table shows that every component contributes positively to the final overall performance of PAM. The main takeaway from this sensitivity analysis is that this architecture under self-supervised learning is quite unstable and does not allow for many added parameters, which will lead to a decrease in performance. For reference, PAM just adds approximately 7.7 million parameters over SBERT, which accounts for a 7% increase.

C Computational Considerations

In this section, we analyze the computational expense of PAM as a crucial factor to consider for practical use. We compare yet again with SBERT, as it serves as the backbone for PAM and is a commonly used metric to evaluate semantic preservation for

[4]https://huggingface.co/google-bert/bert-base-uncased

paraphrase generation (Sousa and Lopes Cardoso, 2024b; Bandel et al., 2022). We averaged the elapsed time for 1000 runs on a Quadro RTX 8000 GPU. Note that each inference requires the extraction and processing of two AMR graphs.

The average inference time per run of PAM was 0.4892 seconds, while for SBERT, it was 0.0173 seconds. This is a substantial increase mainly due to the AMR extraction, which took an average of 0.4603 seconds, which accounts for almost the totality of the inference time of PAM. The remainder of the time is used on the siamese network inference, whose increase in processing time (compared with SBERT) is negligible.

D More Paraphrastic Examples

Here we have more examples o paraphrastic phenomena (see Table 6), following the experiments in Section 9.

Antonym substitution: The sentences do not qualify as paraphrases since they do not convey the same meaning. Instead, they share a one-way entailment relationship: “Pat ate” entails “Pat did not starve”, but the reverse does not necessarily hold. This distinction is crucial when evaluating the appropriateness of the semantic similarity scores. The score of 60.64 underrepresents the strong semantic overlap between the two sentences, while the higher score of 75.42 is more appropriate for capturing the entailment relationship.

Change of person: Here, the grammatical shift does not alter the sentence meaning in this context. Thus, a perfect score aligns with the expected outcome. This perfect score comes from the same AMR representing both sentences:

```
(s / say-01
  :ARG0 (p / person
        :name (n / name
              :op1 "Pat"))
  :ARG1 (l / like-01
        :ARG0 p
        :ARG1 (f / football)))
```

Repetition/Ellipsis: The sentences share significant semantic overlap but are not perfectly equivalent. The first sentence implies an active demonstration, while the second sentence makes a general statement about the demo’s quality. This subtle shift in focus introduces a minor semantic difference, which justifies a slightly lower paraphrase score of 91.81.

Function word variation: The sentences describe the same event, differing only in grammatical structure (the use of function words and syntactic arrangement). This makes them near-perfect paraphrases. The score of 91.81, while high, appropriately accounts for the subtle shift in emphasis from the action performed by Pat to the quality of the demo itself. In contrast, the score of 94.55 may overestimate their similarity, as it overlooks this nuanced semantic distinction.

Phenomenon	Paraphrase (PAM/SBERT)	Non-Paraphrase (PAM/SBERT)
Antonym substitution	Pat ate. $\Leftrightarrow$ Pat did not starve (60.64/**75.42**)	Pat ate. $\not\Leftrightarrow$ Pat did not eat (80.25/**74.67**)
Change of person	Pat said, “I like football.” $\Leftrightarrow$ Pat said that he liked football (**100.0**/97.16)	Pat said, “I like football.” $\not\Leftrightarrow$ Pat heard, “I like football.” (**88.45**/98.15)
Repetition/Ellipsis	Pat can run fast and Chris can run fast, too. $\Leftrightarrow$ Pat can run fast and Chris can, too (**99.45**/98.70)	Pat can run fast and Chris can run fast, too. $\not\Leftrightarrow$ Pat can run fast and Chris is slow (**88.59**/90.52)
Function word variations	Pat showed a nice demo. $\Leftrightarrow$ Pat’s demo was nice (**91.81**/94.55)	Pat showed a nice demo. $\not\Leftrightarrow$ Pat’s demo was ugly (**62.89**/73.06)

Table 6: More paraphrase phenomena examples with paraphrase and non-paraphrase counterparts, including similarity scores. *Paraphrase* pairs are retrieved from the work of Bhagat and Hovy (2013). We highlight the scores we deem best in **bold**.