# Towards a Deeper Understanding of Semantic Comprehension in Language Models Paired with Semantic Graphs

**Matthias Kleiner**[*]
ETH Zürich
makleine@ethz.ch

**Ahmet Özüdogru**[*]
ETH Zürich
oahmet@ethz.ch

**David Zollikofer**[*]
ETH Zürich
zdavid@ethz.ch

## Abstract

Pretrained language models providing contextualized representations significantly advanced the state of the art in numerous NLP tasks. Although they excel at syntax based tasks, it is believed that pretrained models could do a better job at capturing semantic information, as has recently been shown in (Wu et al., 2021) by infusing additional semantic information in the form of DM graphs into graph neural networks stacked on top of language models. In this paper, we present a detailed ablation study on semantic understanding in these models and extend them using AMR graphs, a high level semantic meaning representation language. We find that the infusion of additional semantic information has a close to negligible impact on performance compared to the effect the language model already provides. Furthermore, we show that fine-tuning is at least as important as using contextualized embeddings to perform well on semantic comprehension tasks.

## 1 Introduction

Language models that are pretrained on large amounts of non-annotated data have proven themselves to be almost unanimously useful in numerous downstream tasks; from logical entailment over sentiment classification as well as language generation.

Recent research further implies that models trained using masked language modeling can successfully learn good representations for language syntax. Hence, they are particularly useful for tasks that heavily rely on such information, e.g., constituency parsing and dependency relation labeling (Tenney et al., 2019).

However, models such as RoBERTa (Liu et al., 2019) seemingly struggle with capturing the semantic meaning, as it is suggested in probe studies of recent language models (Wu et al., 2021).

To circumvent this, (Wu et al., 2021) proposes infusing additional semantic information into language models. Concretely, using RoBERTa as a starting point, they augment it with a relational graph convolutional network (RGCN) (Schlichtkrull et al., 2018) which is stacked on-top of the transformer. Using DELPH-IN minimal recursion semantics (DM) (Ivanova et al., 2012; Oepen et al., 2014) they build a semantic graph whose nodes are populated with the corresponding contextualized embeddings from the RoBERTa transformer. An illustration of this can be seen in Figure 3.

By design, the DM structure labels relations between words in a sentence. This allows us to capture some semantic meaning whilst preserving the sentence's original form.

We note that this represents a limitation, as other semantic structures which are not constrained by the sentence's form might contain more useful information for a language model.

Our contribution in this paper lies in using abstract meaning representation (AMR), a high level semantic abstraction (Banarescu et al., 2013), instead of DM as used in (Wu et al., 2021) for infusing semantic information into language models. This is based on the hypothesis that AMR graphs contain additional relevant semantic information compared to DM graphs. For this, we adapt the model from (Wu et al., 2021) to use AMR graphs, as well as provide a pipeline to produce AMR graphs for common NLP evaluation tasks.

Adding AMR graphs allows us to perform detailed ablations on semantic understanding in language models paired with a graph neural network on multiple GLUE (Wang et al., 2018) subtasks. Concretely, we investigate the impact of AMR versus DM in a number of different settings: (1) fine-tuning the underlying transformer, (2) freezing transformer weights and (3) using non-

[*] authors contributed equally, names sorted alphabetically

contextualized embeddings. This allows us to gain a deeper understanding of semantic comprehension in language models paired with semantic graphs.

## 2 Background

Probing studies by (Tenney et al., 2019) find that although pre-trained language models have a good syntactic understanding, the semantic understanding of non fine-tuned deep contextualized word embeddings is lacking compared to their fine-tuned counterparts.

This is in line with previous research such as (Wallace et al., 2019), which found that transformers such as BERT express only limited numeracy knowledge and is outperformed by models such as Word2Vec (Mikolov et al., 2013).

The SIFT paper on which we build (Wu et al., 2021) confirms previous findings with their probing studies indicating that semantic understanding in pre-trained language models is indeed lacking compared to their fine-tuned counterpart.

We note that a literature review has not found additional existing research on integrating AMR graphs into RGCN stacked on top of languages models and to our knowledge there are no detailed ablations on the role the transformer as well as the graph neural network have when it comes to adding semantic information in the form of graphs to language models. Previous research (Wu et al., 2021) does not uncover the complete roles the transformer and the graph neural network play in semantic understanding.

## 3 Methodology

### 3.1 Data Preprocessing

Our original data comes from the GLUE dataset and is explained in detail in Section 4.1. From our original data we have to generate semantic graphs used in the RGCN stacked on top of the language model.

#### 3.1.1 DM

As the CoNLL 2019 shared task pipeline is not publicly accessible, we had to directly use the preprocessed DM graphs as provided by (Wu et al., 2021). We were unable to reproduce them and have confirmed with the CoNLL 2019 shared task organizers that they are not accessible.

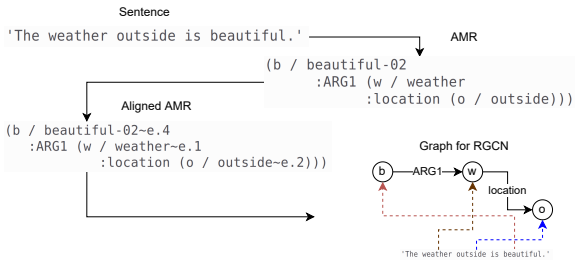


Figure 1: Illustration of our AMR preprocessing pipeline.

#### 3.1.2 AMR

Using AMRLib (Jascob, 2022) we transform our original sentences from the dataset into AMR graphs in Penman format (Goodman, 2020). Note that this requires a specially trained BART model, which generates the AMR graphs (Lewis et al., 2019). The model used by us was trained by AMRLib contributors and has a SMATCH score (Cai and Knight, 2013) of 82.3 on the AMR-3 test set (Knight, 2020) and allows for fast inference. The current state of the art reaches a SMATCH score of 84.9 on the AMR-3 test set (Lam et al., 2021).

As AMR abstracts the semantics of a sentence, there are no longer direct alignments between AMR graph nodes and parts of a sentence, as we had with DM. Since we are populating our graph with embeddings from the transformer, we need to align the words with the nodes. For this, we use AMRLib's built in *Rule Based Word Aligner* to assign tokens to the individual nodes of the AMR graph.

We note that this alignment is likely a weak point in our model, as rule based word alignment has an F1 score of 71.22 on the LDC2014T12 test set (Knight, 2014). We suspect the current state of the art (Cabot et al., 2022) in ARM alignment would have yielded better results, as it outperforms all previous alignment techniques by a large margin. Unfortunately, we were not able to use it as the paper is currently under review as a correspondence with the author has revealed.

The process of transforming a sentence into an aligned AMR graph is detailed in Figure 1.

During our experiments, we witnessed that some AMR graph nodes were not aligned with any words or word-parts from the original sentence. There are two reasons why this happens, (1) the aligner fails to correctly align, or (2) an AMR node stands for a concept (such as location or time etc.). As our model currently fills nodes without an alignment with a zero-embedding, we created a second

```
(f / fly-01~e.1
   :ARG1 (s / she~e.0)
   :destination (c / city
                   :name (n / name
                            :op1 "London"~e.3)))
```

AMRv0

```
(f / fly-01~e.1
   :ARG1 (s / she~e.0)
   :destination (c / city
                   :name (n / name
                            :op1 "London"~e.3)))
```

AMRv1

Figure 2: Example of AMRv0 vs AMRv1 where the concepts `city` and `name` are given non-contextualized embeddings in AMRv1 vs. zero-embedding in AMRv0.

alignment process that builds on the first. The two AMR alignment procedures are named AMRv0 and AMRv1 respectively.

- AMRv0: We use the above-mentioned rule based word aligner to align words from the sentence to our AMR graph. Non-aligned nodes will be populated with a zero-embedding.

- AMRv1: Directly builds on AMRv0, but gives all AMR nodes a corresponding word. Non-aligned nodes have an associated AMR concept. We extract the string representation of the aforementioned concept and assign the AMR node the concept's embedding using RoBERTa's embedding layer, which represents the vocabulary. As the concept is not necessarily part of the sentence, we do not have a contextualized embedding for it, hence we resort to non-contextualized embeddings.

This process is illustrated in Figure 2.

### 3.2 Model Design

In order to gain a deeper understanding of semantic comprehension in language models paired with semantic graphs, we modify the code of the SIFT model from the (Wu et al., 2021) paper and add new training modes, giving us a total of three training modes (completely independent of whether we use AMR or DM).

Note that the following models use the exact same evaluation pipeline as used in SIFT, meaning we max-pool the node embeddings at the end and perform classification on that.

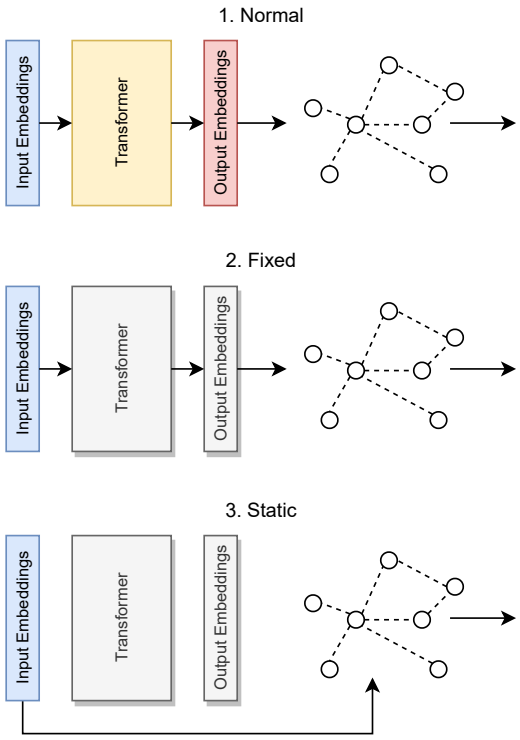


Figure 3: Illustration of the three modes we are evaluating (the graph is either AMR or DM). The weights of the non coloured components are frozen and hence they are not being trained.

- **Normal** Training Mode: This is the SIFT model from (Wu et al., 2021) that uses a transformer paired with a RGCN (Schlichtkrull et al., 2018) with either a DM or an AMR graph extracted from the prompt given. The transformer as well as the RGCN are trained jointly.

- **Fixed** Transformer Training Mode: Analogous to the *Normal Training Mode*, but the transformers weights are frozen and only the RGCN is trained. This investigates the role of the transformer fine-tuning.

- **Static** Embeddings Training Mode: In this case, we use the embeddings from RoBERTa's embedding layer containing the vocabulary. This means we completely bypass the transformer and hence do not have contextualization. This investigates the role of the transformer as a whole and the contextualization that comes with it.

A qualitative illustration of the three models below can be seen in Figure 3.

## 4 Experiments

### 4.1 Datasets

We evaluate all our models on the GLUE (Wang et al., 2018) subtasks RTE, QNLI and MNLI (reporting accuracies on ID as well as OOD). Our work builds upon previous work by (Wu et al., 2021) as well as the 2019 CoNLL shared task. As the data processing pipeline for the 2019 shared task is not publicly available (see Subsection 3.1.1), we were not able to run ablations on other datasets such as HANS (McCoy et al., 2019).

### 4.2 Experimental Setup

All our results build upon a forked version of the original code supplied by (Wu et al., 2021). Besides minor rewrites for compatibility with updated libraries, we run the original SIFT code in the results section 4.4.

Our code as well as instructions on how to reproduce our results can be found on GitHub [1].

As our computing infrastructure was limited, we ran an individual run on a single GPU and used the RoBERTa base models. All experiments were conducted three times (except for MNLI as well as QNLI-static and QNLI-fixed, due to limited resources). In the results section below, we report mean accuracy including standard deviation.

We did not perform a hyperparameter sweep due to the high compute load this would have resulted in. Hence, we used the same hyperparameters as used in the SIFT repository [2], which gave us competitive results even when using AMR graphs. However, we want to note that the reported numbers in Subsection 4.4 are probably biased towards the original SIFT implementation, as the hyperparameters were tuned exclusively for that.

### 4.3 Baselines

As we are actively building upon (Wu et al., 2021) we choose the SIFT-base model as our baseline. [3]

Nevertheless, trying to gain a deeper understanding of semantic comprehension in language models paired with semantic graphs, we perform a qualitative investigation into the role of AMR potentially giving more semantic information compared to DM, contextualization of embeddings, as well as fine-tuning of the transformer.

[1] https://github.com/davidrzs/SemanticsNLPProject

[2] https://github.com/ZhaofengWu/SIFT

[3] RGCN fed with DM on top of RoBERTa, both being trained.

### 4.4 Results

We report the results of our experiments with AMRv0, AMRv1 and DM in normal, fixed transformer and static embedding training modes on the aforementioned datasets in Table 1. Borrowing the representation from the SIFT paper, we highlighted the models whose confidence interval lies above the other models'. For a more detailed explanation, please refer to the description of the Table 1.

**Reproducing SIFT Results** In Table 2 we compare the reproduced results of the original SIFT(Wu et al., 2021) model, in our terms the DM model in normal training mode, with the results from the SIFT paper. Furthermore, we also provide results of RoBERTa without any infusion of semantic information, as given in (Wu et al., 2021). We note that we fail to match the results claimed in the SIFT paper, but our SIFT DM reproduction has a gap smaller than 2 percent to the claimed results.

### 4.5 Analysis & Discussion

First we will look closer at the result of our ablation study by comparing the performance of our models in three different training modes and of RoBERTA without any semantic infusion. At the end we will comment on the implications of using AMR instead of DM graphs for semantic information infusion.

**Impact of Training Modes** RoBERTa without any semantic infusion performs at lest 15 percent better than all of our models in the static embedding training mode (where only RGCN is trained). Since in the static embedding training mode we are not using any language model, this demonstrates that the contribution in capturing semantic meaning of the RGCN is very small compared to the contribution of the language model.

Next, we investigate which part of a language model contributes to the success in capturing semantics. Is it the general ability of the pre-trained transformer, even when weights are fixed, to generate useful contextualized embeddings or is it the careful fine-tuning process giving us more useful contextualized embeddings?

We can answer this question by comparing the results of our different training modes. Observe that all of our models in the fixed encoder training mode perform at least 6 percent better than the corresponding ones in the static embedding training mode. The difference between those training modes is that in the static one we do not use any

| | | | MNLI | |
|---|---|---|---|---|
| **Models** | **RTE** | **QNLI** | **ID.** | **OOD.** |
| **DM** | $79.66_{\pm 0.55}$ | $92.61_{\pm 0.01}$ | $87.15_{\pm 0.10}$ | $87.09_{\pm 0.04}$ |
| **AMRv0** | $79.18_{\pm 2.17}$ | $\mathbf{92.89}_{\pm 0.13}$ | $87.15_{\pm 0.01}$ | $87.20_{\pm 0.08}$ |
| **AMRv1** | $79.18_{\pm 2.17}$ | $92.74_{\pm 0.14}$ | $87.13_{\pm 0.08}$ | $86.89_{\pm 0.01}$ |
| **DM static embeddings** | $50.78_{\pm 0.21}$ | $70.53_{\pm 0.23}$ | $\mathbf{70.20}_{\pm 0.18}$ | $\mathbf{70.19}_{\pm 0.02}$ |
| **AMRv0 static embeddings** | $\mathbf{52.95}_{\pm 0.91}$ | $69.70_{\pm 0.01}$ | $67.46_{\pm 0.13}$ | $67.52_{\pm 0.18}$ |
| **AMRv1 static embeddings** | $\mathbf{53.55}_{\pm 0.42}$ | $69.60_{\pm 0.23}$ | $66.92_{\pm 0.16}$ | $67.63_{\pm 0.30}$ |
| **DM fixed encoder** | $61.61_{\pm 2.11}$ | $84.66_{\pm 0.23}$ | $\mathbf{79.04}_{\pm 0.23}$ | $\mathbf{79.27}_{\pm 0.08}$ |
| **AMRv0 fixed encoder** | $59.93_{\pm 2.01}$ | $84.47_{\pm 0.14}$ | $77.59_{\pm 0.01}$ | $78.34_{\pm 0.20}$ |
| **AMRv1 fixed encoder** | $61.13_{\pm 0.55}$ | $84.41_{\pm 0.11}$ | $77.58_{\pm 0.05}$ | $78.19_{\pm 0.00}$ |

Table 1: Analogous to the SIFT paper, we report mean ± standard deviation; for each bold entry of the DM or AMR model, the corresponding mean minus the standard deviation is no worse than the corresponding mean, of the opposite AMR or DM, plus standard deviation.

| | | | MNLI | |
|---|---|---|---|---|
| **Models** | **RTE** | **QNLI** | **ID.** | **OOD.** |
| **RoBERTa** by (Wu et al., 2021) | $79.0_{\pm 1.6}$ | $93.0_{\pm 0.3}$ | $87.7_{\pm 0.2}$ | $87.3_{\pm 0.3}$ |
| **SIFT DM** by (Wu et al., 2021) | $81.0_{\pm 1.4}$ | $93.2_{\pm 0.2}$ | $87.9_{\pm 0.2}$ | $87.7_{\pm 0.1}$ |
| **SIFT DM Reproduced** | $79.7_{\pm 0.6}$ | $92.6_{\pm 0.0}$ | $87.2_{\pm 0.1}$ | $87.1_{\pm 0.4}$ |

Table 2: Comparison of claimed results and reproduced results. Note that the reported results for RoBERTa and SIFT DM are copied from the SIFT paper.

contextualized embeddings. So we can conclude that this improvement is likely the result of the contextualized embeddings.

Likewise, our models in normal training mode perform at least 7 percent better than the corresponding ones in fixed encoder training mode. This 7 percent increase in accuracy can only be attributed to the finetuning of the model, as it is the only difference between the training modes.

Note that depending on the dataset and the model, the increase in accuracy is not always 6 or 7 percent, but can be much more. However, looking at the results, one can say that both using the contextualized embeddings, and finetuning the weights of RoBERTa play a significant role in capturing semantic information as their respective performance improvements are consistent throughout all tests conducted.

What also caught our attention is that on the QNLI and MNLI datasets, the accuracy in the fixed training mode peaks very quickly in 4 epochs, in static embedding mode in about 10 epochs and in the normal training mode we see a constant improvement over the epochs. We think this is the case because in the static embedding mode the embeddings of the graph nodes are far from optimal, and hence it takes more time for the model to converge in comparison to the fixed training mode, where the embeddings are likely better positioned for data extraction as they are contextualized. In the normal training mode however, we need a lot more epochs for model to converge, we hypothesize that this is due to the model being able to learn more as it has vastly more parameters.

However, we do not observe the same phenomena for the RTE dataset. This might be because the RTE dataset is much smaller compared to the other two.

**Impact of Semantic Representation** Comparing our different models and taking Table 1 into account we can conclude the following:

The most significant case where AMR models perform better than the DM is in the static embedding training mode on the RTE dataset. They achieve an accuracy 2 percent better than the DM model.

On the MNLI dataset, DM dominates AMR models by 2-3 percent in static and fixed training modes, but while almost having the same performance in the normal training mode.

These results are inconclusive, and hence we cannot confirm our hypothesis that AMR graphs

contain additional relevant semantic information compared to DM graphs. Depending on the training mode and the dataset DM or AMR models can perform better.

Furthermore, we add that AMRv0 and AMRv1 perform similarly hinting that the additional information in AMRv1 was not useful to the models learning.

## 5 Conclusion

We have clearly demonstrated that the impact of additional semantic information infusion is minor if not negligible in comparison to the effect the transformer itself has.

Between the three different representations (DM, AMRv0, AMRv1) it is unclear which one provides most utility to our model as their performance ranges are very similar and often overlap. We hypothesize that this directly follows from the fact that the transformer does the heavy lifting and that the RGCN is not able to extract further useful information aiding performance.

We note that we were unable to fully reproduce the results of the SIFT model. In Table 2 one can see that we failed to reproduce the performance claimed in the SIFT (Wu et al., 2021) leading to the reported RoBERTa results actually outperforming it.

## 6 Future Work

We performed all our analyses with the RoBERTa base models and did not perform any hyperparameter optimization. We believe that marginal improvements are possible.

Another possible alternative approach could be to use two parallel paths in a model model where one is a transformer and the other is one utilizing the AMR structure. We hypothesize that a limitation we have is that the embeddings the RGCN builds upon the information in the embeddings the transformer has been able to extract with the limited computing power of the RGCN - perhaps a parallel instead of sequential model could extract more information.

## References


Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Martha Palmer, and Nathan Schneider. 2013. Abstract meaning representation for sembanking. In *In Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, pages 178–186.

Pere-Lluís Huguet Cabot, Abelardo Carlos Martínez Lorenzo, and Roberto Navigli. 2022. Amr alignment: Paying attention to cross-attention.

Shu Cai and Kevin Knight. 2013. Smatch: an evaluation metric for semantic feature structures. volume 2, pages 748–752.

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.

Angelina Ivanova, Stephan Oepen, Lilja Øvrelid, and Dan Flickinger. 2012. Who did what to whom? a contrastive study of syntacto-semantic dependencies. In *Proceedings of the Sixth Linguistic Annotation Workshop*, pages 2–11, Jeju, Republic of Korea. Association for Computational Linguistics.

Brad Jascob. 2022. AMRLib. [Online; accessed 22. Apr. 2022].

Kevin et al. Knight. 2014. Abstract meaning representation (amr) annotation release 1.0.

Kevin et al. Knight. 2020. Abstract meaning representation (amr) annotation release 3.0.

Hoang Thanh Lam, Gabriele Picco, Yufang Hou, Young-Suk Lee, Lam M. Nguyen, Dzung T. Phan, Vanessa López, and Ramón Fernandez Astudillo. 2021. Ensembling graph predictions for AMR parsing. *CoRR*, abs/2110.09131.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *arXiv*.

R Thomas McCoy, Ellie Pavlick, and Tal Linzen. 2019. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. *arXiv preprint arXiv:1902.01007*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, 26.

Stephan Oepen, Marco Kuhlmann, Yusuke Miyao, Daniel Zeman, Dan Flickinger, Jan Hajič, Angelina Ivanova, and Yi Zhang. 2014. SemEval 2014 task 8: Broad-coverage semantic dependency parsing. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 63–72, Dublin, Ireland. Association for Computational Linguistics.

Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *European semantic web conference*, pages 593–607. Springer.

Ian Tenney, Patrick Xia, Berlin Chen, Alex Wang, Adam Poliak, R Thomas McCoy, Najoung Kim, Benjamin Van Durme, Samuel R Bowman, Dipanjan Das, et al. 2019. What do you learn from context? probing for sentence structure in contextualized word representations. *arXiv preprint arXiv:1905.06316*.

Eric Wallace, Yizhong Wang, Sujian Li, Sameer Singh, and Matt Gardner. 2019. Do nlp models know numbers? probing numeracy in embeddings.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*.

Zhaofeng Wu, Hao Peng, and Noah A Smith. 2021. Infusing finetuning with semantic dependencies. *Transactions of the Association for Computational Linguistics*, 9:226–242.