Infusing Finetuning with Semantic Dependencies

Zhaofeng Wu♠ Hao Peng♠ Noah A. Smith♠♦
♠Paul G. Allen School of Computer Science & Engineering, University of Washington
♦Allen Institute for Artificial Intelligence
{zfw7,hapeng,nasmith}@cs.washington.edu

Abstract
For natural language processing systems, two kinds of evidence support the use of text representations from neural language models “pretrained” on large unannotated corpora: performance on application-inspired benchmarks (Peters et al., 2018, inter alia), and the emergence of syntactic abstractions in those representations (Tenney et al., 2019, inter alia). On the other hand, the lack of grounded supervision calls into question how well these representations can ever capture meaning (Bender and Koller, 2020). We apply novel probes to recent language models—specifically focusing on predicate-argument structure as operationalized by semantic dependencies (Ivanova et al., 2012)—and find that, unlike syntax, semantics is not brought to the surface by today’s pretrained models. We then use convolutional graph encoders to explicitly incorporate semantic parses into task-specific finetuning, yielding benefits to natural language understanding (NLU) tasks in the GLUE benchmark. This approach demonstrates the potential for general-purpose (rather than task-specific) linguistic supervision, above and beyond conventional pretraining and finetuning. Several diagnostics help to localize the benefits of our approach.

1 Introduction
The past decade has seen a paradigm shift in how NLP systems are built, summarized as follows:
• Before, general-purpose linguistic modules (e.g., part-of-speech taggers, word-sense disambiguators, and many kinds of parsers) were constructed using supervised learning from linguistic datasets. These were often applied as preprocessing to text as part of larger systems for information extraction, question answering, and other applications.
• Today, general-purpose representation learning is carried out on large, unannotated corpora—effectively a kind of unsupervised learning known as “pretraining”—and then the representations are “finetuned” on application-specific datasets using conventional end-to-end neural network methods.

The newer paradigm encourages an emphasis on corpus curation, scaling up pretraining, and translation of end-user applications into trainable “tasks,” purporting to automate most of the labor requiring experts (linguistic theory construction, annotation of data, and computational model design). Apart from performance improvements on virtually every task explored in the NLP literature, a body of evidence from probing studies has shown that pretraining brings linguistic abstractions to the surface, without explicit supervision (Liu et al., 2019a; Tenney et al., 2019; Hewitt and Manning, 2019; Goldberg, 2019, inter alia).

There are, however, reasons to pause. First, some have argued from first principles that learning mappings from form to meaning is hard from forms alone (Bender and Koller, 2020). Second, probing studies have focused more heavily on syntax than on semantics (i.e., mapping of forms to abstractions of meaning intended by people speaking in the world). Tenney et al. (2019) noted that the BERT model (Devlin et al., 2019) offered more to syntactic tasks like constituent and dependency relation labeling than semantic ones like Winograd coreference and semantic proto-role labeling. Liu et al. (2019a) showed that pretraining did not provide much useful information for entity labeling.

1https://github.com/ZhaofengWu/SIFT

2In fact, Bender and Koller (2020) argued that this is impossible for grounded semantics. Our probing analysis, along with recent efforts (Kovaleva et al., 2019; Liu et al., 2019a), suggests that modern pretrained models are poor at surfacing predicate-argument semantics.
or coreference resolution. Kovaleva et al. (2019) found minimal evidence that the BERT attention heads capture FrameNet (Baker et al., 1998) relations. We extend these findings in §3, showing that representations from the RoBERTa model (Liu et al., 2019b) are relatively poor at surfacing information for a predicate-argument semantic parsing probe, compared to what can be learned with finetuning, or what RoBERTa offers for syntactic parsing. The same pattern holds for BERT.

Based on that finding, we hypothesize that semantic supervision may still be useful to tasks targeting natural language “understanding.” In §4, we introduce semantics-infused finetuning (SIFT), inspired by pre-neural pipelines. Input sentences are first passed through a semantic dependency parser. Though the method can accommodate any graph over tokens, our implementation uses the DELPH-IN MRS-derived dependencies, known as “DM” (Ivanova et al., 2012), illustrated in Figure 1. The task architecture learned during finetuning combines the pretrained model (here, RoBERTa) with a relational graph convolutional network (RGCN; Schlichtkrull et al., 2018) that reads the graph parse. Though the same graph parser can be applied at inference time (achieving our best experimental results), benefits to task performance are in evidence in a “light” model variant without inference time parsing and with the same inference cost as a RoBERTa-only baseline.

We experiment with the GLUE benchmarks (§5), which target many aspects of natural language understanding (Wang et al., 2018). Our model consistently improves over both base and large sized RoBERTa baselines.³ Our focus is not on achieving a new state of the art, but we note that SIFT can be applied orthogonally alongside other methods that have improved over similar baselines, such as Raffel et al. (2020) and Clark et al. (2020) which used alternative pretraining objectives, and Jiang et al. (2020) which proposed an alternative finetuning optimization framework. In §6, we use the HANS (McCoy et al., 2019) and GLUE (Wang et al., 2018) diagnostics to better understand where our method helps on natural language inference tasks. We find that our model’s gains strengthen when finetuning data is reduced, and that our approach is more effective than alternatives that do not use the full labeled semantic dependency graph.

## 2 Predicate-Argument Semantics as Dependencies

Though many formalisms and annotated datasets have been proposed to capture various facets of natural language semantics, here our focus is on predicates and arguments evoked by words in sentences. Our experiments focus on the DELPH-IN dependencies formalism (Ivanova et al., 2012), commonly referred to as “DM” and derived from minimal recursion semantics (Copestake et al., 2005) and head-driven phrase structure grammar (Pollard and Sag, 1994). This formalism, illustrated in Figure 1 (top, blue) has the appealing property that a sentence’s meaning is represented as a labeled, directed graph. Vertices are words (though not every word is a vertex), and 59 labels are used to characterize argument and adjunct relationships, as well as conjunction.

Other semantic formalisms such as PSD (Hajic et al., 2012), EDS (Oepen and Lønning, 2006), and UCCA (Abend and Rappoport, 2013) also capture semantics as graphs. Preliminary experiments showed similar findings using these. Frame-based predicate-argument representations such as those found in PropBank (Palmer et al., 2005) and FrameNet (Baker et al., 1998) are not typically cast as graphs (rather as “semantic role

³RoBERTa-base and RoBERTa-large use the same pretraining data and only differ in the number of parameters.
labeling”), but see Surdeanu et al. (2008) for data transformations and Peng et al. (2018b) for methods that help bridge the gap.

Graph-based formalizations of predicate-argument semantics, along with organized shared tasks on semantic dependency parsing (Oepen et al., 2014, 2015), enabled the development of data-driven parsing methods following extensive algorithm development for dependency syntax (Eisner, 1996; McDonald et al., 2005). Even before the advent of the pretraining-finetuning paradigm, labeled $F_1$ scores above 0.9 were achieved (Peng et al., 2017).

Some similarities between DM and dependency syntax (e.g., the Stanford dependencies, illustrated in Figure 1, bottom, red; de Marneffe et al., 2006) are apparent: both highlight bilexical relationships. However, semantically empty words (like infinitival to) are excluded from the semantic graph, allowing direct connections between semantically related pairs (e.g., technique ← apply, impossible → apply, and apply → crops, all of which are mediated by other words in the syntactic graph). DM analyses need not be trees as in most syntactic dependency representations, so they may more directly capture the meaning of many constructions, such as control.

3 Probing RoBERTa for Predicate-Argument Semantics

The methodology known as “linguistic probing” seeks to determine the level to which a pretrained model has rediscovered a particular linguistic abstraction from raw data (Shi et al., 2016; Adi et al., 2017; Hupkes et al., 2018; Belinkov and Glass, 2019, inter alia). The procedure is:

1. Select an annotated dataset that encodes the theoretical abstraction of interest into a predictive task, usually mapping sentences to linguistic structures. Here we will consider the Penn Treebank (Marcus et al., 1993) converted to Stanford dependencies and the DM corpus from CoNLL 2015’s shared task 18 (Oepen et al., 2015).

2. Pretrain. We consider RoBERTa and BERT.

3. Train a full-fledged “ceiling” model with finetuned representations. It can be seen as proxy to the best performance one can get with the pretrained representations.

4. Train a supervised “probe” model for the task with the pretrained representations. Importantly, the pretrained representations should be frozen, and the probe model should be lightweight with limited capacity, so that its performance is attributable to pretraining. We use a linear probe classifier.

5. Compare, on held-out data, the probe model against the ceiling model. Through such a comparison, we can estimate the extent to which the pretrained model “already knows” how to do the task, or, more precisely, brings relevant features to the surface for use by the probing model.

Liu et al. (2019a) included isolated DM arc prediction and labeling tasks and Tenney et al. (2019) conducted “edge probing.” To our knowledge, full-graph semantic dependency parsing has not been formulated as a probe.

For both syntactic and semantic parsing, our full ceiling model and our probing model are based on the Dozat and Manning (2017, 2018) parser which underlies many state-of-the-art systems (Clark et al., 2018; Li et al., 2019, inter alia). Our ceiling model contains nonlinear multilayer perceptron (MLP) layers between RoBERTa/BERT and the arc/label classifiers, as in the original parser, and finetunes the pretrained representations. The probing model, trained on the same data, freezes the representations and removes the MLP layers, yielding a linear model with limited capacity. We measure the conventionally reported metrics: labeled attachment score for dependency parsing and labeled $F_1$ for semantic parsing, as well as labeled and unlabeled exact match scores. We follow the standard practice and use the Chu-Liu-Edmonds algorithm (Chu and Liu, 1965; Edmonds, 1967) to decode the syntactic dependency trees and greedily decode the semantic graphs with local edge/label classification decisions. See Appendix B for training details.

Comparisons between absolute scores on the two tasks are less meaningful. Instead, we are interested in the difference between the probe (largely determined by pretrained representations) and the ceiling (which benefits also from finetuning). Prior work leads us to expect that the se-
4 Finetuning with Semantic Graphs

Given pretrained RoBERTa’s relative incapability of surfacing semantic structures (§3) and the importance of modeling predicate-argument semantics (§2), we hypothesize that incorporating such information into the RoBERTa finetuning process should benefit downstream NLU tasks.

SIFT, briefly outlined in §4.1, is based on the relational graph convolutional network (RGCN; Schlichtkrull et al., 2018). §4.2 introduces a lightweight variant of SIFT aiming to reduce test time memory and runtime.

4.1 SIFT

SIFT first uses an external parser to get the semantic analysis for the input sentence. Then it contextualizes the input with a pretrained RoBERTa model, the output of which is fed into a graph encoder building on the semantic parse. We use RGCN to encode the DM structures, which are labeled graphs. The model is trained end-to-end. Figure 2 diagrams this procedure.

RGCN. RGCN can be understood as passing vector “messages” among vertices in the graph. The nodes are initially represented with RoBERTa token embeddings. At each RGCN layer, each node representation is updated with a learned composition function. For tasks such as text classification or regression, we max-pool over the final RGCN layer’s output to get a sequence-level representation for onward computation. Readers are referred to Appendix A and Schlichtkrull et al. (2018) for further details.

Note on tokenization. RoBERTa uses byte-pair encodings (BPE; Sennrich et al., 2016), differing from the standard BPE used in SIFT, briefly outlined in §4.1, aiming to reduce test time memory and runtime.

Table 1: The RoBERTa-base (top) and RoBERTa-large (bottom) parsing results for the full ceiling model and the probe on the PTB Stanford Dependencies (SD) test set and CoNLL 2015 in-domain test set. We also report their absolute and relative differences (probe – full). The smaller the magnitude of the difference, the more relevant content the pretrained model already encodes. We report the canonical parsing metric (LAS for PTB dependency and labeled F1 for DM) and labeled/unlabeled exact match scores (LEM/UEM). All numbers are mean ± standard deviation across three seeds.

<table>
<thead>
<tr>
<th>Metrics</th>
<th>Abs Δ</th>
<th>Rel Δ</th>
<th>Ceiling</th>
<th>Probe</th>
</tr>
</thead>
<tbody>
<tr>
<td>PTB SD</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LAS/F1</td>
<td>-13.5±0.2</td>
<td>-14.2±0.2</td>
<td>95.2±0.1</td>
<td>81.7±0.1</td>
</tr>
<tr>
<td>LEM</td>
<td>-36.4±0.8</td>
<td>-72.4±1.1</td>
<td>50.3±0.5</td>
<td>13.9±0.5</td>
</tr>
<tr>
<td>UEM</td>
<td>-46.3±0.7</td>
<td>-73.2±0.5</td>
<td>63.2±0.8</td>
<td>17.0±0.3</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Metrics</th>
<th>Abs Δ</th>
<th>Rel Δ</th>
<th>Ceiling</th>
<th>Probe</th>
</tr>
</thead>
<tbody>
<tr>
<td>CoNLL 2015 DM</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Abs Δ</td>
<td>-23.5±0.1</td>
<td>-24.9±0.2</td>
<td>94.2±0.0</td>
<td>70.7±0.2</td>
</tr>
<tr>
<td>Rel Δ</td>
<td>-45.4±1.1</td>
<td>-93.5±0.5</td>
<td>48.5±1.2</td>
<td>3.1±0.2</td>
</tr>
<tr>
<td>Ceiling</td>
<td>-48.8±1.0</td>
<td>-92.8±0.5</td>
<td>52.6±1.0</td>
<td>3.8±0.2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Metrics</th>
<th>Abs Δ</th>
<th>Rel Δ</th>
<th>Ceiling</th>
<th>Probe</th>
</tr>
</thead>
<tbody>
<tr>
<td>PTB SD</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LAS/F1</td>
<td>-17.6±0.1</td>
<td>-18.5±0.1</td>
<td>95.3±0.0</td>
<td>77.7±0.1</td>
</tr>
<tr>
<td>LEM</td>
<td>-40.0±0.6</td>
<td>-77.2±0.4</td>
<td>51.9±0.6</td>
<td>11.8±0.2</td>
</tr>
<tr>
<td>UEM</td>
<td>-50.2±0.6</td>
<td>-77.4±0.2</td>
<td>64.8±0.7</td>
<td>14.6±0.2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Metrics</th>
<th>Abs Δ</th>
<th>Rel Δ</th>
<th>Ceiling</th>
<th>Probe</th>
</tr>
</thead>
<tbody>
<tr>
<td>CoNLL 2015 DM</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Abs Δ</td>
<td>-26.7±0.3</td>
<td>-28.3±0.3</td>
<td>94.4±0.1</td>
<td>67.7±0.2</td>
</tr>
<tr>
<td>Rel Δ</td>
<td>-46.6±1.1</td>
<td>-94.4±0.1</td>
<td>49.3±1.1</td>
<td>2.7±0.0</td>
</tr>
<tr>
<td>Ceiling</td>
<td>-50.0±1.1</td>
<td>-93.9±0.2</td>
<td>53.2±1.0</td>
<td>3.3±0.1</td>
</tr>
</tbody>
</table>

Note on tokenization. RoBERTa uses byte-pair encodings (BPE; Sennrich et al., 2016), differing from the standard BPE used in SIFT, aiming to reduce test time memory and runtime.
4.2 SIFT-Light

Inspired by the scaffold model of Swayamdipta et al. (2018), we introduce SIFT-Light, a lightweight variant of SIFT that aims to reduce time and memory overhead at test time. During inference it does not rely on explicit semantic structures and therefore has the same computational cost as the RoBERTa baseline.

SIFT-Light learns two classifiers (or regressors): (1) a main linear classifier on top of RoBERTa \( f_{\text{RoBERTa}} \); (2) an auxiliary classifier \( f_{\text{RGCN}} \) based on SIFT. They are separately parameterized at the classifier level, but share the same underlying RoBERTa. They are trained on the same downstream task and jointly update the RoBERTa model. At test time, we only use \( f_{\text{RoBERTa}} \). The assumption behind SIFT-Light is similar to the scaffold framework of Swayamdipta et al. (2018): by sharing the RoBERTa parameters between the two classifiers, the contextualized representations steer towards downstream classification with semantic encoding. One key difference is that SIFT-Light learns with two different architectures for the same task, instead of using the multitask learning framework of Swayamdipta et al. (2018). In §6.3, we find that SIFT-Light outperforms a scaffold.

4.3 Discussion

Previous works have used GCN (Kipf and Welling, 2016), a similar architecture, to encode unlabeled syntactic structures (Marcheggiani and Titov, 2017; Bastings et al., 2017; Zhang et al., 2020c,a, inter alia). We use RGCN to explicitly encode labeled semantic graphs. Our analysis shows that it outperforms GCN, as well as alternatives such as multitask learning with parameter-sharing (§6.3). However, this comes with a cost. In RGCN, the number of parameters linearly increases with the number of relation types. In our experiments, on top of the 125M RoBERTa-base parameters, this adds approximately 3–118M parameters to the model, depending on the hyperparameter settings (see Appendix B). On top of RoBERTa-large, which itself has 355M parameters, this adds 6–121M additional parameters. The inference runtime of SIFT is 1.41–1.79 \( \times \) RoBERTa’s with the base size and 1.30–1.53 \( \times \) with the large size.

SIFT incorporates semantic information only during finetuning. Recent evidence suggests that structural information can be learned with specially-designed pretraining procedures. For example, Swayamdipta et al. (2019) pretrain with syntactic chunking, requiring the entire pretraining corpus to be parsed which is computationally prohibitive at the scale of RoBERTa’s pretraining dataset. With a distillation technique, Kuncoro et al. (2020) bake syntactic supervision into the pretraining objective. Despite better accuracy on tasks that benefit from syntax, they show that the

---

6In experiments we upper-bound the number of the parameters by imposing a low-rank constraint on the parameter matrices by construction. See Appendix A.
obtained syntactically-informed model hurts the performance on other tasks, which could restrict its general applicability. Departing from these alternatives, SIFT augments general-purpose pretraining with task-specific structural finetuning, an attractively modular and flexible solution.

5 Experiments

We next present experiments with SIFT to test
our hypothesis that pretrained models for natural
language understanding tasks benefit from explicit
predicate-argument semantics.

5.1 Settings

We use the GLUE datasets, a suite of tests tar-
geting natural language understanding detailed in
Table 2 (Wang et al., 2018). Most are class-
ification datasets, with STS-B considers regres-
sion. Among the classifications datasets, MNLI
has three classes while others have two; CoLA and
SST-2 classify single sentences while the rest clas-
sify sentence pairs. We follow Dodge et al. (2020)
and Vu et al. (2020) and only report development
set results due to restricted GLUE test set access.

We compare the following models:

• RoBERTa, both the base and large variants, following Liu et al. (2019b).

• SIFT builds on pretrained RoBERTa, with 2
RGCN layers. To generate semantic graphs, we use the semantic dependency parser by
Che et al. (2019) which held the first place in the CoNLL 2019 shared task (Oepen et al.,
2019) with 92.5 labeled F1 for DM. SIFT-Light ($\S$4.2) is trained similarly to
SIFT, but does not rely on inference-time parsing.

• Syntax-infused finetuning is similar to SIFT but uses the syntactic Universal Depen-
dencies parser (Straka, 2018; Straka and Straková, 2019) from the CoNLL 2019
shared task (Oepen et al., 2019). We include this model to confirm that any benefits to task
performance are due specifically to the se-
mic structures.

Hyperparameters are summarized in Appendix B.

Implementation details. We run all models
across 3 seeds for the large datasets QNLI,
MNLI, and QQP (due to limited computational
resources), and 4 seeds for all others. As we
do not aim for state of the art, we do not
use intermediate task training, ensemble models, or
re-formulate QNLI as a ranking task as done by
Liu et al. (2019b). For sentence-pair classification
tasks such as MNLI, we use structured decompos-
able attention (Parikh et al., 2016) and 2 additional
RGCN layers to further propagate the attended in-
formation (Chen et al., 2017). The two graphs are
separately max-pooled to obtain the final represen-
tation. See Appendix A for more details.

5.2 Main Findings

Tables 3 summarizes the GLUE development set
performance of the four aforementioned models
when they are implemented with RoBERTa-base
and RoBERTa-large. With RoBERTa-base (Table
3a), SIFT achieves a consistent improvement
over the baseline across the board, suggesting
that despite heavy pretraining, RoBERTa still ben-
efits from explicit semantic structural informa-
tion. Among the datasets, smaller ones tend to
obtain larger improvements from SIFT, e.g., 1.7
Matthews correlation for CoLA and 2.0 accuracy
for RTE, while the gap is smaller on the larger
one (e.g., only 0.1 accuracy for QQP). Moreover,

\footnote{About half of the CoNLL 2019 evaluation set is out-of-
domain. Without gold semantic graph annotations for our
target datasets, this can be seen as a reasonable estimation of
the parser’s performance for our use case.}
We show ablation results for architectural scenarios in which incorporating semantic structures helps RoBERTa. In this section, we first analyze how SIFT-Light often improves over RoBERTa, with a smaller gap, making it a compelling model choice when latency is prioritized. This shows that encoding semantics using RGCN is not only capable of producing better standalone output representations, but can also benefit the finetuning of the RoBERTa-internal weights through parameter sharing. Finally, the syntax-infused model underperforms SIFT across all tasks. It only achieves minor improvements over RoBERTa, if not hurting performance. These results provide evidence supporting our hypothesis that incorporating semantic structures is more beneficial to RoBERTa than syntactic ones.

We observe a similar trend with RoBERTa-large in Table 3b, where SIFT’s absolute improvements are very similar to those in Table 3a. Specifically, both achieve a 0.6 accuracy improvement over RoBERTa, averaged across all datasets. This indicates that the increase from RoBERTa-base to RoBERTa-large added little to surfacing semantic information.

6 Analysis and Discussion

In this section, we first analyze in which scenarios incorporating semantic structures helps RoBERTa. We then highlight SIFT’s data efficiency and compare it to alternative architectures. We show ablation results for architectural decisions in Appendix D. All analyses are conducted on RoBERTa-base.

6.1 When Do Semantic Structures Help?

Using two diagnostic datasets designed for evaluating and analyzing natural language inference models, we find that SIFT (1) helps guard the model against frequent but invalid heuristics in the data, and (2) better captures nuanced sentence-level linguistic phenomena than RoBERTa.

Results on the HANS diagnostic data. We first diagnose the model using the HANS dataset (McCoy et al., 2019). It aims to study whether a natural language inference (NLI) system adopts three heuristics, summarized and exemplified in Table 4. The premise and the hypothesis have high surface form overlap, but the heuristics are not valid for reasoning. Each heuristic has both positive and negative (i.e., entailment and non-entailment) instances constructed. Due to the high surface similarity, many models tend to predict “entailment” for the vast majority of instances. As a result, they often reach decent accuracy on the entailment examples, but struggle on the “non-entailment” ones (McCoy et al., 2019), on which we focus our analysis. The 30,000 test examples are evenly spread among the 6 classes (3 heuristics, 2 labels). Table 4 compares SIFT against the RoBERTa

<table>
<thead>
<tr>
<th>Models</th>
<th>CoLA</th>
<th>MRPC</th>
<th>RTE</th>
<th>SST-2</th>
<th>STS-B</th>
<th>QNLI</th>
<th>QQP</th>
<th>ID.</th>
<th>OOD.</th>
<th>Avg.</th>
</tr>
</thead>
<tbody>
<tr>
<td>RoBERTa</td>
<td>63.1±0.9</td>
<td>90.1±0.8</td>
<td>79.0±1.6</td>
<td>94.6±0.3</td>
<td>91.0±0.0</td>
<td>93.0±0.3</td>
<td>91.8±0.1</td>
<td>87.7±0.2</td>
<td>87.3±0.3</td>
<td>86.4</td>
</tr>
<tr>
<td>SIFT</td>
<td>64.8±0.4</td>
<td>90.5±0.7</td>
<td>81.0±1.4</td>
<td>95.1±0.4</td>
<td>91.3±0.1</td>
<td>93.2±0.2</td>
<td>91.9±0.1</td>
<td>87.9±0.2</td>
<td>87.7±0.1</td>
<td>87.0</td>
</tr>
<tr>
<td>SIFT-Light</td>
<td>64.1±1.3</td>
<td>90.3±0.5</td>
<td>80.6±1.4</td>
<td>94.7±0.1</td>
<td>91.2±0.1</td>
<td>92.8±0.3</td>
<td>91.7±0.0</td>
<td>87.7±0.1</td>
<td>87.6±0.1</td>
<td>86.7</td>
</tr>
<tr>
<td>Syntax</td>
<td>63.5±0.6</td>
<td>90.4±0.5</td>
<td>80.9±1.0</td>
<td>94.7±0.5</td>
<td>91.1±0.2</td>
<td>92.8±0.2</td>
<td>91.8±0.0</td>
<td>87.9±0.1</td>
<td>87.7±0.1</td>
<td>86.7</td>
</tr>
</tbody>
</table>

Table 3: GLUE development set results with RoBERTa-base (top) and RoBERTa-large (bottom). We report Matthews correlation for CoLA, Pearson’s correlation for STS-B, and accuracy for others. We report mean ± standard deviation; for each bold entry, the mean minus standard deviation is no worse than RoBERTa’s corresponding mean plus standard deviation.

SIFT-Light often improves over RoBERTa, with a smaller gap, making it a compelling model choice when latency is prioritized. This shows that encoding semantics using RGCN is not only capable of producing better standalone output representations, but can also benefit the finetuning of the RoBERTa-internal weights through parameter sharing. Finally, the syntax-infused model underperforms SIFT across all tasks. It only achieves minor improvements over RoBERTa, if not hurting performance. These results provide evidence supporting our hypothesis that incorporating semantic structures is more beneficial to RoBERTa than syntactic ones.
<table>
<thead>
<tr>
<th>Heuristic</th>
<th>Premise</th>
<th>Hypothesis</th>
<th>Label</th>
<th>RoBERTa</th>
<th>SIFT</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Lexical Overlap</strong></td>
<td>The banker near the judge saw the actor. The judge by the actor stopped the banker.</td>
<td>The banker saw the actor.</td>
<td>E</td>
<td>98.3</td>
<td>98.9</td>
</tr>
<tr>
<td></td>
<td>The judge by the actor stopped the banker.</td>
<td>The banker stopped the actor.</td>
<td>N</td>
<td>68.1</td>
<td>71.0</td>
</tr>
<tr>
<td><strong>Sub-sequence</strong></td>
<td>The artist and the student called the judge. The judges heard the actors resigned.</td>
<td>The student called the judge.</td>
<td>E</td>
<td>99.7</td>
<td>99.8</td>
</tr>
<tr>
<td></td>
<td>The judges heard the actors resigned.</td>
<td>The judges heard the actors.</td>
<td>N</td>
<td>25.8</td>
<td>29.5</td>
</tr>
<tr>
<td><strong>Constituent</strong></td>
<td>Before the actor slept, the senator ran. If the actor slept, the judge saw the artist.</td>
<td>The actor slept.</td>
<td>E</td>
<td>99.3</td>
<td>98.8</td>
</tr>
<tr>
<td></td>
<td>The actor slept.</td>
<td></td>
<td>N</td>
<td>37.9</td>
<td>37.6</td>
</tr>
</tbody>
</table>

Table 4: HANS heuristics and RoBERTa-base and SIFT’s accuracy. Examples are due to McCoy et al. (2019). “E”: entailment. “N”: non-entailment. Bold font indicates better result in each category.

<table>
<thead>
<tr>
<th>Phenomenon</th>
<th>RoBERTa</th>
<th>SIFT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Predicate Argument Structure</td>
<td>43.5</td>
<td>44.6</td>
</tr>
<tr>
<td>Logic</td>
<td>36.2</td>
<td>38.3</td>
</tr>
<tr>
<td>Lexical Semantics</td>
<td>45.6</td>
<td>44.8</td>
</tr>
<tr>
<td>Knowledge</td>
<td>28.0</td>
<td>26.3</td>
</tr>
</tbody>
</table>

Table 5: $R_3$ correlation coefficient of RoBERTa-base and SIFT on the GLUE diagnostic set.

baseline on HANS. Both struggle with non-entailment examples. SIFT yields improvements on the lexical overlap and subsequence heuristics, which we find unsurprising, given that semantic analysis directly addresses the underlying differences in meaning between the (surface-similar) premise and hypotheses in these cases. SIFT performs similarly to RoBERTa on the constituent heuristic with a 0.3% accuracy difference for the non-entailment examples. Here the hypothesis corresponds to a constituent in the premise, and therefore we expect its semantic parse to often be a subgraph of the premise’s; accuracy hinges on the meanings of the connectives (e.g., before and if in the examples), not on the structure of the graphs.

Results on the GLUE diagnostic data. GLUE’s diagnostic set (Wang et al., 2018) contains 1,104 artificially-curated NLI examples to test a model’s performance on various linguistic phenomena including predicate-argument structure (e.g., “I opened the door.” entails “The door opened.” but not “I opened.”), logic (e.g., “I have no pet puppy.” entails “I have no corgi pet puppy.” but not “I have no pets.”), lexical semantics (e.g., “I have a dog.” entails “I have an animal.” but not “I have a cat.”), and knowledge & common sense (e.g., “I went to the Grand Canyon.” entails “I went to the U.S.” but not “I went to Antarctica.”). Table 5 presents the results in $R_3$ correlation coefficient (Gorodkin, 2004). Explicit semantic dependencies help SIFT perform better on predicate-argument structure and sentence logic. On the other hand, SIFT underperforms the baseline on lexical semantics and world knowledge. We would not expect a benefit here, since semantic graphs do not add lexical semantics or world knowledge; the drop in performance suggests that some of what RoBERTa learns is lost when it finetuned through sparse graphs. Future work might seek graph encoding architectures that mitigate this loss.

6.2 Sample Efficiency
In §5.2, we observe greater improvements from SIFT on smaller finetuning sets. We hypothesize that the structured inductive bias helps SIFT more when the amount of finetuning data is limited. We test this hypothesis on MNLI by training different models varying the amount of finetuning data. We train all configurations with the same three random seeds. As seen in Table 6, SIFT offers larger improvements when less finetuning data is used.

Given the success of the pretraining paradigm, we expect many new tasks to emerge with tiny finetuning sets, and these will benefit the most from methods like SIFT.

6.3 Comparisons to Other Graph Encoders
In this section we compare RGCN to some commonly used graph encoders. We aim to study whether or not (1) encoding graph labels helps, and (2) explicitly modeling discrete structures is necessary. Using the same experiment setting as in §5.1, we compare SIFT and SIFT-Light to

- Graph convolutional network (GCN; Kipf and Welling, 2016). GCN does not encode relations, but is otherwise the same as RGCN.
- Graph attention network (GAT; Veličković...
<table>
<thead>
<tr>
<th>Fraction</th>
<th></th>
<th>Train</th>
<th>RoBERTa</th>
<th>SIFT</th>
<th>Abs Δ</th>
<th>Rel Δ</th>
<th>RoBERTa</th>
<th>SIFT</th>
<th>Abs Δ</th>
<th>Rel Δ</th>
</tr>
</thead>
<tbody>
<tr>
<td>100%</td>
<td>392k</td>
<td>87.7</td>
<td>87.9</td>
<td>0.2</td>
<td>0.2%</td>
<td>87.3</td>
<td>87.7</td>
<td>0.4</td>
<td>0.4%</td>
<td></td>
</tr>
<tr>
<td>0.5%</td>
<td>1,963</td>
<td>76.1</td>
<td>77.6</td>
<td>1.5</td>
<td>1.9%</td>
<td>77.1</td>
<td>78.2</td>
<td>1.1</td>
<td>1.4%</td>
<td></td>
</tr>
<tr>
<td>0.2%</td>
<td>785</td>
<td>68.6</td>
<td>71.0</td>
<td>2.5</td>
<td>3.5%</td>
<td>70.0</td>
<td>71.8</td>
<td>1.8</td>
<td>2.5%</td>
<td></td>
</tr>
<tr>
<td>0.1%</td>
<td>392</td>
<td>58.7</td>
<td>61.2</td>
<td>2.6</td>
<td>4.2%</td>
<td>60.5</td>
<td>63.7</td>
<td>3.3</td>
<td>5.1%</td>
<td></td>
</tr>
</tbody>
</table>

Table 6: RoBERTa-base and SIFT’s performance on the entire MNLI development sets and their absolute and relative differences, with different numbers of finetuning instances randomly subsampled from the training data.

<table>
<thead>
<tr>
<th>Models</th>
<th>CoLA</th>
<th>MRPC</th>
<th>RTE</th>
<th>SST-2</th>
<th>STS-B</th>
<th>QNLI</th>
<th>QQP</th>
<th>ID.</th>
<th>OOD.</th>
<th>Avg.</th>
</tr>
</thead>
<tbody>
<tr>
<td>RoBERTa</td>
<td>63.1</td>
<td>90.1</td>
<td>79.0</td>
<td>94.6</td>
<td>91.0</td>
<td>93.0</td>
<td>91.8</td>
<td>87.7</td>
<td>87.3</td>
<td>86.4</td>
</tr>
<tr>
<td>GCN</td>
<td><strong>65.2</strong></td>
<td>90.2</td>
<td>80.2</td>
<td>94.8</td>
<td>91.1</td>
<td>92.9</td>
<td>91.8</td>
<td>87.8</td>
<td>87.7</td>
<td>86.8</td>
</tr>
<tr>
<td>GAT</td>
<td>63.4</td>
<td>90.0</td>
<td>79.4</td>
<td>94.7</td>
<td>91.2</td>
<td>92.9</td>
<td>91.8</td>
<td>87.7</td>
<td>87.6</td>
<td>86.5</td>
</tr>
<tr>
<td>Hidden</td>
<td>64.2</td>
<td>90.2</td>
<td>79.7</td>
<td>94.5</td>
<td>91.0</td>
<td>92.8</td>
<td>91.8</td>
<td>87.1</td>
<td>86.7</td>
<td>86.4</td>
</tr>
<tr>
<td>Scaffold</td>
<td>62.5</td>
<td><strong>90.5</strong></td>
<td>71.1</td>
<td>94.3</td>
<td>91.0</td>
<td>92.6</td>
<td>91.7</td>
<td>87.7</td>
<td>87.6</td>
<td>85.5</td>
</tr>
<tr>
<td>SIFT</td>
<td>64.8</td>
<td><strong>90.5</strong></td>
<td><strong>81.0</strong></td>
<td><strong>95.1</strong></td>
<td><strong>91.3</strong></td>
<td><strong>93.2</strong></td>
<td><strong>91.9</strong></td>
<td><strong>87.9</strong></td>
<td><strong>87.7</strong></td>
<td><strong>87.0</strong></td>
</tr>
<tr>
<td>SIFT-Light</td>
<td>64.1</td>
<td>90.3</td>
<td>80.6</td>
<td>94.7</td>
<td>91.2</td>
<td>92.8</td>
<td>91.7</td>
<td>87.7</td>
<td>87.6</td>
<td>86.7</td>
</tr>
</tbody>
</table>

Table 7: GLUE development set results for different architectures for incorporating semantic information. The settings and metrics are identical to Table 3a. All models use the base size variant.

et al., 2018). Similarly to GCN, it encodes unlabeled graphs. Each node aggregates representations of its neighbors using an attention function (instead of convolutions).

- **Hidden** (Pang et al., 2019; Zhang et al., 2020a). It does not explicitly encode structures, but uses the hidden representations from a pretrained parser as additional features to the classifier.

- **Scaffold** (Swayamdipta et al., 2018) is based on multitask learning. It aims to improve the downstream task performance by additionally training the model on the DM data with a full parsing objective.

To ensure fair comparisons, we use comparable implementations for these models. We refer the readers to the works cited for further details.

Table 7 summarizes the results, with SIFT having the highest average score across all datasets. Notably, the 0.2 average absolute benefit of SIFT over GCN and 0.5 over GAT demonstrates the benefit of including the semantic relation types (labels). Interestingly, on the linguistic acceptability task—which focuses on well-formedness and therefore we expect relies more on syntax—GCN outperforms RGCN-based SIFT. GAT underperforms GCN by 0.3 on average, likely because the sparse semantic structures (i.e., small degrees of each node) make attended message passing less useful. Hidden does not on average outperform the baseline, highlighting the benefit of discrete graph structures (which it lacks). Finally, the scaffold underperforms across most tasks.

7 Related Work

Using explicit linguistic information. Before pretrained contextualized representations emerged, linguistic information was commonly incorporated into deep learning models to improve their performance including part of speech (Sennrich and Haddow, 2016; Xu et al., 2016, *inter alia*) and syntax (Eriguchi et al., 2017; Chen et al., 2017; Miwa and Bansal, 2016, *inter alia*).

Nevertheless, recent attempts in incorporating syntax into pretrained models have little success on NLU: Strubell et al. (2018) found syntax to only marginally help semantic role labeling with ELMo, and Kuncoro et al. (2020) observed that
incorporating syntax into BERT conversely hurts the performance on some GLUE NLU tasks. On the other hand, fewer attempts have been devoted to incorporating sentential predicate-argument semantics into NLP models. Zhang et al. (2020b) embedded semantic role labels from a pretrained parser to improve BERT. However, these features do not constitute full sentential semantics. Peng et al. (2018a) enhanced a sentiment classification model with DM but only used one-hop information and no relation modeling.

**Probing syntax and semantics in models.** Many prior works have probed the syntactic and semantic content of pretrained transformers, typically BERT. Wallace et al. (2019) observed that BERT displays suboptimal numeracy knowledge. Clark et al. (2019) discovered that BERT’s attention heads tend to surface syntactic relationships. Hewitt and Manning (2019) and Tenney et al. (2019) both observed that BERT embeds a significant amount of syntactic knowledge. Besides pretrained transformers, Belinkov et al. (2020) used syntactic and semantic dependency relations to analyze machine translation models.

**8 Conclusion**

We presented strong evidence that RoBERTa and BERT do not bring predicate-argument semantics to the surface as effectively as they do for syntactic dependencies. This observation motivates SIFT, which aims to incorporate explicit semantic structures into the pretraining-finetuning paradigm. It encodes automatically parsed semantic graphs using RGCN. In controlled experiments, we find consistent benefits across eight tasks targeting natural language understanding, relative to RoBERTa and a syntax-infused RoBERTa. These findings motivate continued work on task-independent semantic analysis, including training methods that integrate it into architectures serving downstream applications.

**Acknowledgments**

The authors thank the anonymous reviewers for feedback that improved the paper. We also thank Stephan Oepen for help in producing the CoNLL 2019 shared task companion data, Yutong Li for contributing to early experiments, and Elizabeth Clark and Lucy Lin for their suggestions and feedback. This research was supported in part by a Google Fellowship to HP and NSF grant 1562364.

**References**


Wanxiang Che, Longxu Dou, Yang Xu, Yuxuan Wang, Yijia Liu, and Ting Liu. 2019. HIT-SCIR
at MRP 2019: A unified pipeline for meaning representation parsing via efficient training and effective encoding. In Proc. of MRP.


Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. ELECTRA: Pre-training text encoders as discriminators rather than generators. In Proc. of ICLR.

Kevin Clark, Minh-Thang Luong, Christopher D. Manning, and Quoc Le. 2018. Semi-supervised sequence modeling with cross-view training. In Proc. of EMNLP.


Olga Kovaleva, Alexey Romanov, Anna Rogers, and Anna Rumshisky. 2019. Revealing the dark secrets of BERT. In *Proc. of EMNLP*.


Deric Pang, Lucy H. Lin, and Noah A. Smith. 2019. Improving natural language inference


Hao Peng, Sam Thomson, and Noah A. Smith. 2018a. Backpropagating through structured argmax using a SPIGOT. In *Proc. of ACL.*

Hao Peng, Sam Thomson, Swabha Swayamdipta, and Noah A. Smith. 2018b. Learning joint semantic parsers from disjoint data. In *Proc. of NAACL.*


Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proc. of EMNLP.*


Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proc. ACL.*

Xing Shi, Inkit Padhi, and Kevin Knight. 2016. Does string-based neural MT learn source syntax? In *Proc. of EMNLP.*

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proc. of EMNLP.*


Milan Straka and Jana Straková. 2019. ÚFAL MRPipe at MRP 2019: UDPipe goes semantic in the meaning representation parsing shared task. In *Proc. of MRP.*


context? Probing for sentence structure in contextualized word representations. In Proc. of ICLR.


Kun Xu, Siva Reddy, Yansong Feng, Songfang Huang, and Dongyan Zhao. 2016. Question answering on freebase via relation extraction and textual evidence. In Proc. of ACL.


A Detailed Model Architecture

In this section we provide a detailed illustration of our architecture.

Graph Initialization Because RoBERTa’s BPE tokenization differs from the Che et al. (2019) semantic parser’s CoNLL 2019 tokenization, we align the two tokenization schemes using character level offsets, as illustrated in Figure 3. For each node $i$, we find wordpieces $[t_j, \cdots , t_k]$ that it aligns to. We initialize its node embedding by averaging the vectors of these wordpiece followed by a learned affine transformation and a ReLU nonlinearity:

$$h_i^{(0)} = \text{ReLU} \left( \frac{1}{k-j+1} \sum_{s=j}^{k} e_s \right)$$

Here $W_e$ is a learned matrix, and the $e$ vectors are the wordpiece representations. The superscript on $h$ denotes the layer number, with 0 being the input embedding vector fed into the RGCN layers.

Graph Update In each RGCN layer $\ell$, every node’s hidden representation is propagated to its direct neighbors:

$$h_i^{(\ell+1)} = \text{ReLU} \left( \sum_{r \in R} \sum_{j \in \mathcal{N}_i^r} \frac{1}{|\mathcal{N}_i^r|} W_r^{(\ell)} h_j^{(\ell)} + W_0^{(\ell)} h_i^{(\ell)} \right)$$

where $R$ is the set of all possible relations (i.e., edge labels; including inverse relations for inverse edges that we manually add corresponding to the original edges) and $\mathcal{N}_i^r$ denotes $i$’s neighbors with relation $r$. $W_r$ and $W_0$ are learned parameters representing a relation-specific transformation and a self-loop transformation, respectively. We also use the basis-decomposition trick described in Schlichtkrull et al. (2018) to reduce the number of parameters and hence the memory requirement. Specifically, we construct $B$ basis matrices; where $|R| > B$, the transformation of each relation is constructed by a learned linear combination of the basis matrices. Each RGCN layer captures the neighbors information that is one hop away. We use $\ell = 2$ RGCN layers for our experiments.

Sentence Pair Tasks For sentence pair tasks, it is crucial to model sentence interaction (Parikh et al., 2016). We therefore use a similar structured decomposable attention component to model the interaction between the two semantic graphs. Each node attends to the other graph’s nodes using biaffine attention; its output is then concatenated to its node representation calculated in its own graph. Specifically, for two sentences $a$ and $b$, we obtain an updated representation $h_{i,a}^{(\ell)}$ for $a$ as follows:

$$\alpha_{i,j} = \text{biaffine} \left( h_i^{(\ell,a)} , h_j^{(\ell,b)} \right)$$

$$\hat{h}_{i,a}^{(\ell)} = \sum_{j} \alpha_{i,j} h_j^{(\ell,b)}$$

$$h_{i,a}^{(\ell)} = \text{ReLU} \left( W_\alpha \left[ h_i^{(\ell,a)} ; \hat{h}_{i,a}^{(\ell)} ; h_i^{(\ell,\hat{a})} - \hat{h}_{i,a}^{(\ell)} ; h_i^{(\ell,a)} \odot \hat{h}_{i,a}^{(\ell,a)} \right] \right)$$

where $W_\alpha$ is a learned matrix, and $\odot$ denotes the elementwise product. We do the same operation to obtain the updated $h_{i,b}^{(\ell,b)}$. Inspired by Chen et al. (2017), we add another $\ell$ RGCN composition layers to further propagate the attended representation. They result in additional parameters and runtime cost compared to what was presented in §4.3.

Graph Pooling The NLU tasks we experiment with require one vector representation for each instance. We max-pool over the sentence graph (for sentence pair tasks, separately for the two graphs whose pooled output are then concatenated), concatenate it with RoBERTa’s [CLS] embedding, and feed the result into a layer normalization layer (LN) to get the final output.

Figure 4: SIFT architecture for sentence pair tasks. Two graphs are first separately encoded using RGCN, then structured decomposable attention is used to capture the inter-graph interaction. Additional RGCN layers are used to further propagate the structured information. Finally two vectors max-pooled from both graphs are concatenated and used for onward computation. RoBERTa and the external parser are suppressed for clarity.
Table 8: The BERT-base parsing results for the full ceiling model and the probing model on the PTB Stanford Dependencies (SD) test set and CoNLL 2015 in-domain test set. The metrics and settings are identical to Table 1 except only one seed is used.

<table>
<thead>
<tr>
<th>Metrics</th>
<th>PTB SD</th>
<th>CoNLL 2015 DM</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Abs Δ</td>
<td>Rel Δ</td>
</tr>
<tr>
<td>LAS/F₁</td>
<td>−13.6</td>
<td>−14.4%</td>
</tr>
<tr>
<td>LEM</td>
<td>−35.8</td>
<td>−73.7%</td>
</tr>
<tr>
<td>UEM</td>
<td>−44.7</td>
<td>−74.1%</td>
</tr>
</tbody>
</table>

Table 9: Ablation results on the development sets of 3 GLUE datasets with a RoBERTa-base backbone.

<table>
<thead>
<tr>
<th>Metrics</th>
<th>MNLI</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MRPC</td>
</tr>
<tr>
<td>Full</td>
<td>90.5</td>
</tr>
<tr>
<td>− attention</td>
<td>90.1</td>
</tr>
<tr>
<td>− concat</td>
<td>90.2</td>
</tr>
</tbody>
</table>

C BERT Probing Results

We replicate the RoBERTa probing experiments described in §3 for BERT. We observe similar trends where the probing model degrades more from the full model for DM than dependency syntax. This demonstrates that, like RoBERTa, BERT also less readily surfaces semantic content than syntax.

D Ablations

In this section we ablate two major architectural choices: the sentence pair structured decomposable attention component and the use of a concatenated RoBERTa and RGCN representation rather than only using the latter. We select 3 sentence-pair datasets covering different dataset sizes and tasks with identical experimental setup as §5.1. The ablation results in Table 9 show that the full SIFT architecture performs the best.

B Hyperparameters

Probing Hyperparameters. No hyperparameter tuning is conducted for the probing experiments. For the full models, we use intermediate MLP layers with dimension 512 for arc projection and 128 for label projection. The probing models do not have such layers. We minimize the sum of the arc and label cross entropy losses for both dependency and DM parsing. All models are optimized with AdamW (Loshchilov and Hutter, 2019) for 10 epochs with batch size 8 and learning rate $2 \times 10^{-5}$.

Main Experiment Hyperparameters. For SIFT, we use 2 RGCN layers for single-sentence tasks and 2 additional composition RGCN layers after the structured decomposable attention component for sentence-pair tasks. The RGCN hidden dimension is searched in $\{256, 512, 768\}$, the number of bases in $\{20, 60, 80, 100\}$, dropout between RGCN layers in $\{0, 0.2, 0.3\}$, and the final dropout after all RGCN layers in $\{0, 0.1\}$. For SIFT-Light, the training loss is obtained with $0.2 \text{loss}_{RGCN} + 0.8 \text{loss}_{RoBERTa}$. For all models, the number of training epochs is searched in $\{3, 10, 20\}$ and the learning rate in $\{1 \times 10^{-4}, 2 \times 10^{-5}\}$. We use 0.1 weight decay and 0.06 warmup ratio. All models are optimized with AdamW with an effective batch size of 32.