

6 Appendix

6.1 Models

- BERT: Off the shelf "bert-base-uncased" from the huggingface transformers library (Wolf et al., 2019)
- QA-SQUAD-1: Both SQuAD QA models are trained with the huggingface question answering training script ⁴. This adds a span prediction head to the default BERT, i.e. a linear layer that computes logits for the span start and span end. So for a given question and a context, it classifies the indices in which the answer starts and ends. As a loss function it uses crossentropy. The model was trained on a single GPU. We used the huggingface default training script and standard parameters: 2 epochs, learning rate 3e-5, batch size 12.
- QA-SQUAD-2: Single GPU, also using huggingface training script with standard parameters. Learning rate was 3e-5, batch size 12, best model after 2 epochs.
- MLM-SQUAD: Fine tuned on text from SQUAD using the masked language modeling objective as per (Devlin et al., 2019). 15% of the tokens masked at random. Trained for 4 epochs with LR 5e-5. Single GPU.
- RANK-MSMARCO: Trained as described in (Nogueira and Cho, 2019). MSMARCO, 100k iterations with batch size 128 (on a TPUv3-8).
- MLM-MSMARCO: 15% of the tokens masked at random. 3 epochs, batch size 8, LR 5e-5. Single gpu.

6.2 Experimental results:

- Computing infrastructure used: Everything can be run in Colab notebook with 12gb of RAM and the standard GPU. The experiments, however, have been run on a computing cluster with 6 nodes. Every node had 4 gtx 1080ti and 128gb RAM. Thus being able to parallelize the probing of different layers.

- Average runtime: Circa 3 hours per layer (that is training the MLM head and probing the LAMA probes) on a single GPU.
- Number of parameters: Since we use standard BERT, the base model + MLM head combined have 110,104,890 parameters. The MLM head itself has 24,459,834 parameters.
- Validation performance for test results: Since we probed the data, we could not do validation on it.
- Explanation of evaluation metrics used with links to code: It is done in knowledge_probing/probing/metrics.py. But the one that we use are Precision @ k where we just check if the model predicts the correct token at index $\leq k$ (P@k)

6.3 Hyperparameter search:

Not applicable.

6.4 Datasets:

- Wikitext-2: Used for fine-tuning the MLM head. Subset of the English Wikipedia for long term dependency language modeling. 2,088,628 tokens for training, 217,646 for validation, 245,569 for testing. Vocab size: 33,278 out of vocab: 2.6% of tokens. It can be downloaded from here: <https://www.salesforce.com/products/einstein/ai-research/the-wikitext-dependency-language-modeling-dataset/>
- LAMA probe data: Can be downloaded from their github: <https://github.com/facebookresearch/LAMA>. Only used for testing. Consists of: Google-RE: 5528 instances over 3 relations. T-REx: 34017 instances over 41 relations. ConceptNet: 12514 instances. This is not grouped into relations. Squad: 305 instances. Context in-sensitive questions rewritten to cloze-statements. No specific relation either.
- SQuAD 1.1: Can be downloaded from here: <https://rajpurkar.github.io/SQuAD-explorer/>. 100,000+ question answer pairs based on wikipedia articles. Produced by crowdworkers.
- SQuAD 2: Can be downloaded from here: <https://rajpurkar.github.io/SQuAD-explorer/>

⁴<https://github.com/huggingface/transformers>

- . Combines the 100,000+ question answer pairs with 50,000 unanswerable questions.
- MSMARCO: Can be downloaded from here: <https://microsoft.github.io/msmarco/> . For ranking: Dataset for passage reranking was used. Given 1,000 passages, re-rank by relevance. Dataset contains 8,8m passages. For MLM training: Dataset for QA was used. It consists of over 1m queries and the 8,8m passages. Each query has 10 candidate passages. For MLM, we appended the queries with all candidate passages before feeding into BERT.

6.5 Knowledge captured in BERT

6.5.1 Intermediate Layers Matter

Additional precisions for Figure 4 can be found in Figure 9.

6.5.2 Relational Knowledge Evolution

Additional precisions for Figure 2 can be found in Figure 10.

6.5.3 Effect of dataset size

Figure 11 and 12 show the P@10 and P@100 plots for Figure 5. Respectively, Figure 13 and 14 show the same for 6.

6.6 Effect of fine tuning objective

For comparing MLM and QA on SQuAD (7), Figure 15 and 16 show more precisions. Also, for comparing fine tune objectives on MSMARCO (Figure 8), Figure 17 and 18 show P@10 and P@100.

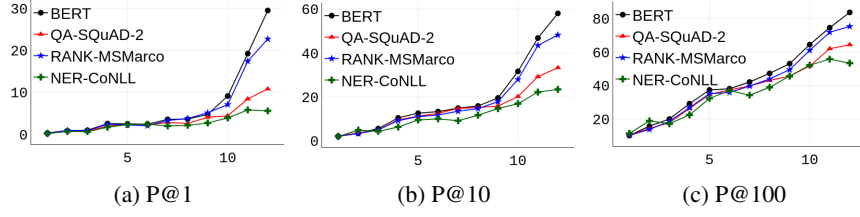


Figure 9: Mean performance in different precisions on T-REx sets for BERT, QA-SQUAD-2, RANK-MSMARCO, NER-CoNLL.

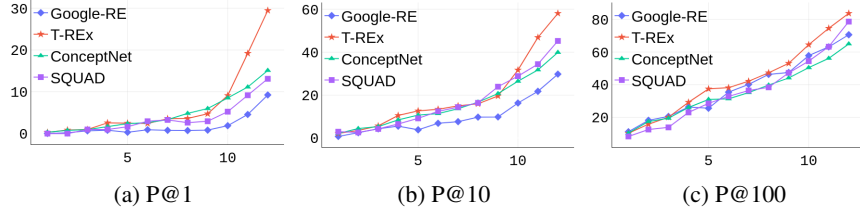


Figure 10: Mean performance of BERT across all layers and probe sets.

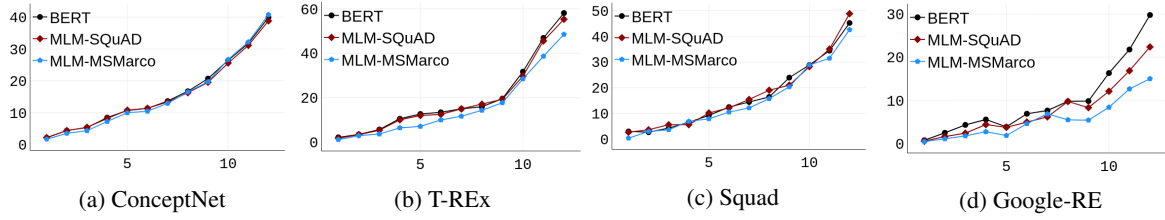


Figure 11: Effect of dataset size. Showing P@10

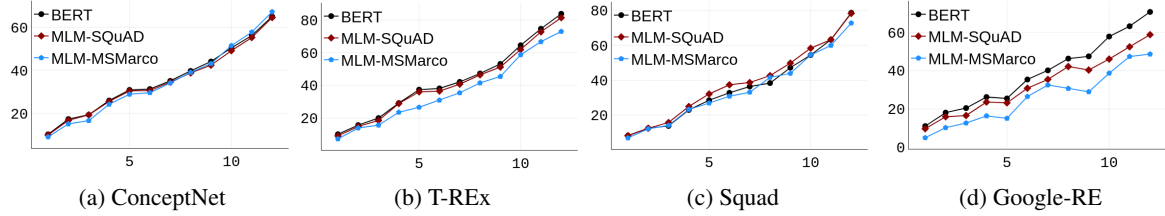


Figure 12: Effect of dataset size. Showing P@100

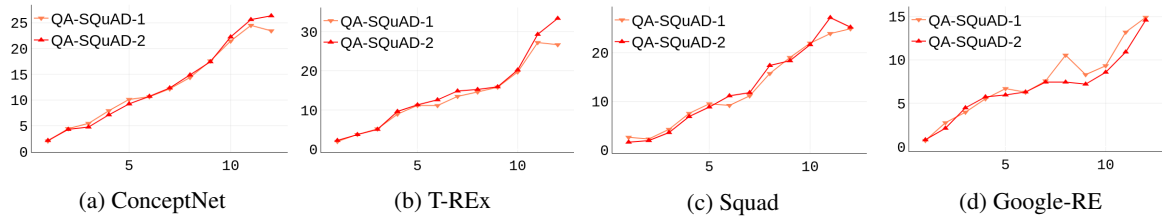


Figure 13: Effect of dataset size. Showing P@10 for the QA objective.

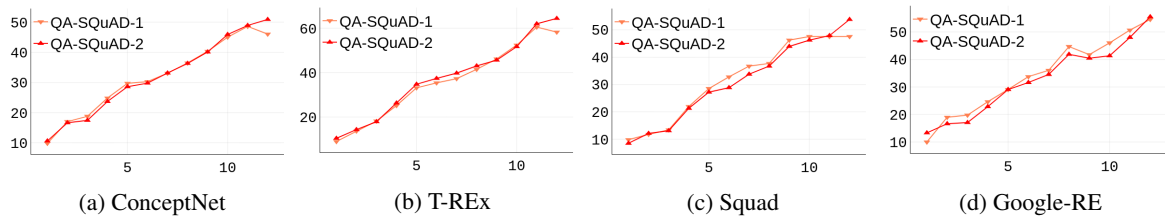


Figure 14: Effect of dataset size. Showing P@100 for the QA objective.

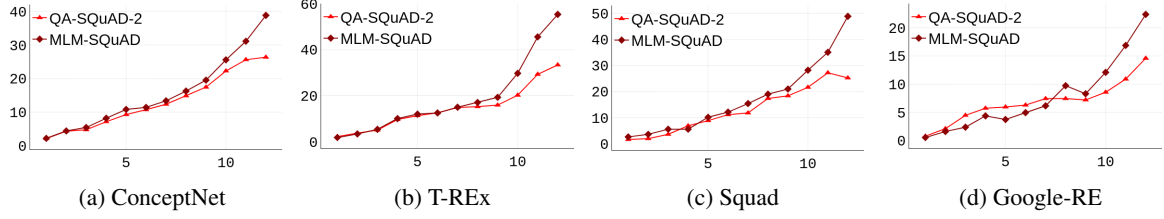


Figure 15: Effect of Fine-Tuning Objective on fixed size data: SQUAD. Showing P@10.

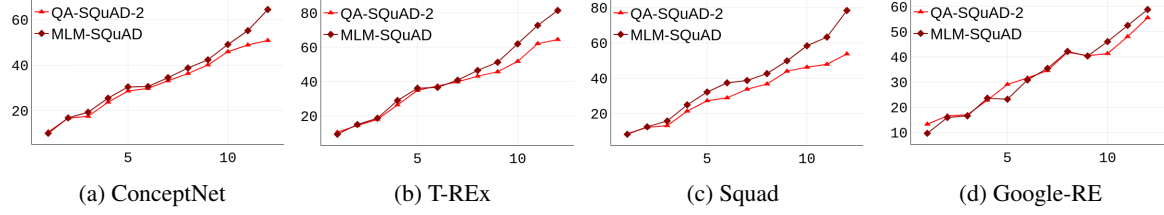


Figure 16: Effect of Fine-Tuning Objective on fixed size data: SQUAD. Showing P@100.

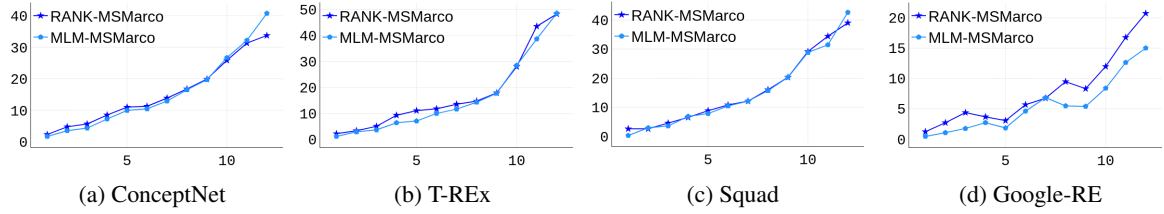


Figure 17: Effect of Fine-Tuning Objective on fixed size data: MSMarco. Showing P@10.

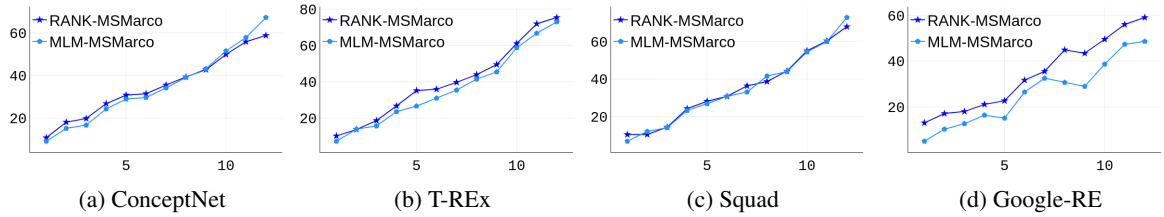


Figure 18: Effect of Fine-Tuning Objective on fixed size data: MSMarco. Showing P@100.

Model	Google-RE		T-REx		ConceptNet		Squad	
	P@1	P@1	P@1	P@1	P@1	P@1	P@1	P@1
BERT	10	15	29	34	15	21	13	20
QA-SQUAD-1	3	9	6	15	7	15	5	15
QA-SQUAD-2	3	9	10	19	8	16	6	13
MLM-SQUAD	4	10	15	23	9	16	6	16
RANK-MSMARCO	6	11	23	29	12	19	10	20
MLM-MSMARCO	3	7	14	21	11	17	7	12

Table 3: Mean knowledge contained in the last layer (P@1) vs knowledge contained in all layers (P@1) for each probe.