Can Edge Probing Tests Reveal Linguistic Knowledge in QA Models?

Sagnik Ray Choudhury[1*, 3], Nikita Bhutani[2], and Isabelle Augenstein[3]

[1]University of Michigan
[2]Megagon Labs
[3]University of Copenhagen

sagnikrayc@gmail.com, nikita@megagon.ai, augenstein@di.ku.dk

Abstract

There have been many efforts to try to understand what grammatical knowledge (e.g., ability to understand the part of speech of a token) is encoded in large pre-trained language models (LM). This is done through 'Edge Probing' (EP) tests: supervised classification tasks to predict the grammatical properties of a span (whether it has a particular part of speech) using *only* the token representations coming from the LM encoder. However, most NLP applications fine-tune these LM encoders for specific tasks. Here, we ask: if an LM is fine-tuned, does the encoding of linguistic information in it change, as measured by EP tests? Specifically, we focus on the task of Question Answering (QA) and conduct experiments on multiple datasets. We find that EP test results do not change significantly when the fine-tuned model performs well or in adversarial situations where the model is forced to learn wrong correlations. From a similar finding, some recent papers conclude that fine-tuning does not change linguistic knowledge in encoders but they do not provide an explanation. We find that EP models themselves are susceptible to exploiting spurious correlations in the EP datasets. When this dataset bias is corrected, we do see an improvement in the EP test results as expected.

1 Introduction

The encoding of linguistic information in large pre-trained language models (LMs) such as BERT (Devlin et al., 2019) has become an active research topic in recent times. This encoding is usually measured by edge probing (EP) tasks (Liu et al., 2019; Tenney et al., 2019a). Consider the sentence "the Met is closing soon". The token 'met' is a noun (a museum and not a form of the verb 'meet'). The context words ('the', 'is') are the only signals for determining its part of speech. If a 'simple' (one or two layer MLP (Hewitt and Liang, 2019)) classifier predicts 'met' as a noun *only* using the representation of the token 'met' (coming from a contextual encoder such as BERT (Devlin et al., 2019) or ELMo (Peters et al., 2018)) and not the whole sentence, then these signals must have been encoded in the token representation itself. This is the grammatical knowledge the test is 'probing' for. If encoder E_1 performs better than encoder E_2 on an EP test, say, part-of-speech tagging, we say that E_1 has a better knowledge of part-of-speech than E_2.

For many NLP tasks, pre-trained LMs (most commonly, BERT) have emerged as standard encoders (Raffel et al., 2020). These encoders are fine-tuned after adding task-specific layers on top. While probing tests on pre-trained encoders are quite popular, fine-tuned encoders are relatively under-explored (with notable exceptions of Merchant et al. (2020) and van Aken et al. (2019). We aim to bridge this gap by probing fine-tuned models using question answering (QA) as a target task. QA is a complex NLU problem requiring the model to implicitly perform many reasoning steps, and fine-tuned models provide strong baselines for various QA datasets (Devlin et al., 2019). Our first research question is thus:

RQ1: *Does fine-tuning for QA tasks improve the encoding of linguistic skills in the encoders, when measured by existing EP tests?* Intuitively, DNN based QA models would require implicit knowledge of semantic roles (who did what to whom, when, and where), an understanding of the part of speech and entity boundaries (most answers are entities in the context), and anaphora resolution (entities in the context would refer to each other). Indeed, prior works show how injecting knowledge about semantic roles (Shen and Lapata, 2007) and coreference resolution (González and Rodríguez, 2000) in classical QA pipelines improves their performance. Therefore, a fine-tuned QA model should *implicitly* acquire these linguistic skills. The

[*] Work done at the University of Copenhagen.

QA layers in the fine-tuned models have much fewer parameters than the encoders (§3), therefore, the encoders themselves are more likely to encode that grammatical knowledge. Consequently, when these fine-tuned encoders are used for the SRL (semantic role labeling), CoREF (coreference), PoS (part-of-speech tagging), and NER (named entity recognition) EP tests, we would expect to see improvements over pre-trained LMs. **But we do not observe any such change (§4)**.

Fine-tuning is generally performed on much less data than pre-training and the encoder weights might not change significantly. Could that cause the EP test results to remain the same? If the encoder weights are kept fixed during fine-tuning, the performance in the target (QA) task drops $50-70\%$ on all datasets. However, this frozen encoder has the same performance on the EP test as the original one. In other words, two encoders with a high difference in the target task performances have no discernible difference in the EP task performances. A possible explanation is that the QA models have no need to use the grammatical knowledge we are testing for. This motivates the second research question:

RQ2: *Does fine-tuning for QA tasks impart the linguistic skills necessary to perform QA in the encoders?* To answer this, we create a QA dataset that requires a particular 'skill' (Rogers et al., 2022; Ray Choudhury et al., 2022): the knowledge of coreference resolution (§5). Quoref (Dasigi et al., 2019) is such a dataset, but one might not require the knowledge of coreference to answer *all* questions in Quoref. Many instances in standard NLU tasks can be solved by heuristics, i.e., without proper *reasoning* (see Gururangan et al. (2018) for NLI or Min et al. (2019) for multi-hop QA). We design algorithms to filter out questions that can be answered heuristically (§5), and consequently, any model *probably* needs to use the knowledge of coreference to answer the rest. However, two encoders with a significant performance difference on this de-biased dataset have no difference in the CoREF EP test. This motivates the third research question:

RQ3: *Why do the EP test results not reflect that encoders have learned the linguistic skills needed to perform QA?*: Our analysis (§6) of the EP test datasets suggests that the EP models themselves might rely on dataset biases (as opposed to learning the task with input representations). When this bias is corrected, fine-tuned encoders behave as expected, i.e., show significant performance improvements over the base encoders. Previously, van Aken et al. (2019) and Merchant et al. (2020) observed that the EP test results do not differ in the base vs fine-tuned encoders[1] ($RQ1$) and concluded that the encoding of grammatical knowledge in the encoders does not change during fine-tuning. However, unlike ours, their studies were not done on the problems that explicitly call for such grammatical knowledge. Moreover, current criticisms of EP tests on non fine-tuned encoders focus on the task design itself (Hewitt and Liang, 2019; Voita and Titov, 2020) (see §6), whereas this work calls for bias correction in the standardized EP test datasets.

2 Related Work

Prior work has focused on understanding various aspects of pre-trained LMs including attention patterns (Clark et al., 2019) and linguistic knowledge (Liu et al., 2019). When these LMs are used as encoders in models, they turn out to be strong baselines for many tasks (Raffel et al., 2020). However, less is known about how the fine-tuning process changes the encoder's attention patterns (Kovaleva et al., 2019) or their encoding of linguistic knowledge. While many papers (Jia and Liang, 2017; Kaushik and Lipton, 2018; Sen and Saffari, 2020; Sugawara et al., 2020, inter alia) argue that DNN models often use heuristics to answer questions, Ray Choudhury et al. (2022) shows that at least some of the models use human-interpretable reasoning steps. Therefore, it is important to study how edge probing tests capture the task-specific reasoning abilities introduced in the fine-tuning process.

The paradigm of the classifier based probing tasks (of which our EP tasks are a subset) is quite mature (Ettinger et al., 2016), and has seen increasing popularity with the release of benchmark EP datasets (the ones we use here) (Tenney et al., 2019a). Typically, internal layers of large language or machine translation models are used as features for auxiliary prediction tasks for syntactic properties: part-of-speech (Shi et al., 2016; Blevins et al., 2018; Tenney et al., 2019a), tense (Shi et al., 2016; Tenney et al., 2019a), or subject-verb agreement

[1]Merchant et al. (2020) uses one QA dataset (SQuAD (Rajpurkar et al., 2016)) and all our EP tests; van Aken et al. (2019) uses two datasets (SQuAD and HotpotQA (Yang et al., 2018)) and two of our EP tests; which makes this study more rigorous in the QA domain.

(Tran et al., 2018; Linzen et al., 2016). See Belinkov and Glass (2019) for an extensive survey.

However, not many papers study the benchmark EP tests for fine-tuned representations. Most similar to our work is the layer-wise analysis of BERT weights for QA (van Aken et al., 2019) and the results of fine-tuning BERT on EP tasks (Merchant et al., 2020) – van Aken et al. (2019) use three QA datasets (SQuAD, HotpotQA, and bAbi (Weston et al., 2016)) to show that: 1) for the EP task of CoREF, test results remain unchanged, even when representations are taken from different layers; 2) different layers of a fine-tuned BERT can be attributed to different tasks in the QA process such as supporting fact extraction or entity selection. Merchant et al. (2020) studies MNLI, dependency parsing, and QA (SQuAD) to arrive at a similar finding, although their main results use a scalar mix of the weights from all layers of a fine-tuned BERT (whereas our work uses the top layer). RQ1 in our work can be considered complementary to their work, but RQ2 and RQ3 have not been studied before.

EP tests are indirect, i.e., a classifier (probe) is used to measure the linguistic information in the representation. Do the test results reflect the quality of the representations or the classifier's ability to learn the task (Hewitt and Liang, 2019; Voita and Titov, 2020)? We discuss this in §6. See Belinkov (2022) for more background on probing classifiers.

3 Edge Probing & QA: The Setup

Edge Probing: Following prior work (Merchant et al., 2020; van Aken et al., 2019), we use the model architecture and four of the edge probing tasks proposed by Tenney et al. (2019a). Given a sentence $S = [T_1, ...T_n]$ of n tokens where $T_i \in \mathcal{R}^d$, for PoS and NER tasks, the goal is to predict the part of speech or entity tag for a set of spans $T_i..T_j, 1 \leq i, j \leq n$ in that sentence (with *only* the span and not the sentence as the input). Using the same setting for SRL and CoREF tasks, the input is a pair of spans and the output is a class label: for SRL, it is a semantic role, for CoREF it is a binary label indicating whether one span is an antecedent of the other or not. A self-attention pooling operator is used to generate a fixed representation for spans of different lengths (Lee et al., 2017). For SRL and CoREF, these representations for the two spans are concatenated. A single-layer linear probe is used for the actual classification task (Figure 1).

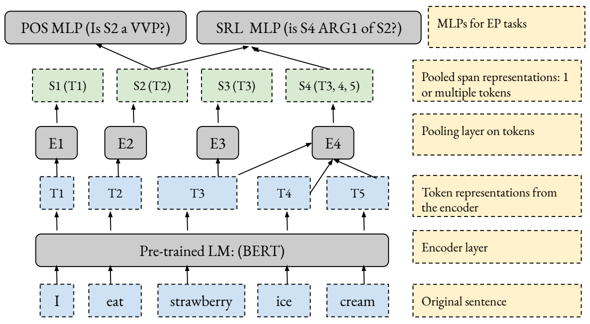

Figure 1: The architecture for edge probing tasks. For all tasks (PoS, SRL, CoREF, NER) the same MLP is used.

For EP tests, the span representations need to be generated from the token representations. Token representations can be generated from each layer (Tenney et al., 2019a) or the top layer (Tenney et al., 2019b) of the encoder. In each case, the layer i representation can be calculated as: 1. *Just* the output of layer i, 2. A concatenation of the first layer output and layer i output ('cat'), and 3. A scalar mixing of the output of $0 - i$ layers where the mixing weights are trainable parameters ('mix'). In all experiments, we use the 'cat' setting for the topmost layer in the encoder because both Merchant et al. (2020) and we find no significant difference for the other settings (the top layers generally perform better).

Question Answering: In typical QA setups, models are given a context and a question as input. We use two span-based datasets (SQuAD, Rajpurkar et al. (2016) and HotpotQA, Yang et al. (2018)) where the task is to extract a span of the text from the context as the answer. We also use two MCQ datasets (ReCoRD, Zhang et al. (2018) and MultiRC, Khashabi et al. (2018)) where the model is trained to select an answer from a set of choices. The architectures follow Devlin et al. (2019) and are similar for all datasets: one or two QA-specific layers on top of an encoder (see the appendix for more details).

4 RQ1: EP Tests & Fine-Tuned Encoders

We first run the EP tests with a standard encoder. Next, the QA models (the same encoder + QA layers) are trained (fine-tuned). Then, in the fine-tuned models the QA layers are replaced with the same MLP layers used in the EP tests, and the tests are repeated. This gives us a measure of how much the encoding of linguistic knowledge in the encoder might have *improved* due to the fine-tuning process. We also randomize the QA training data (for SQuAD and HotpotQA) by using random noun phrases as answers. Thus the model is forced to learn wrong correlations, which can cause it to 'forget' the skills to some extent, and in turn, the encoder should perform worse on the EP tests.

4.1 Experimental Setup

We use a $\text{BERT}_{\text{base-uncased}}$ model as the base encoder for all tasks. For each QA task, we run five experiments. The span-based QA models are evaluated using the F1 score and the exact match (EM) metric. The exact match measures the percentage of answers that exactly match the actual answers. The F1 score measures the token overlap between the predicted and the actual answer. The MCQ questions are classification tasks and evaluated using the usual accuracy and Micro-F1 metrics for classification.

For EP tests we use the highest performing model in each QA dataset. The tests use the benchmark OntoNotes 5.0 corpus (Weischedel et al., 2013), as in Tenney et al. (2019a) and Merchant et al. (2020). We use the same hyper-parameters as the original paper on EP tests (Tenney et al., 2019b) except for the batch size (32 theirs vs 16 ours).[2] The QA models were trained for 10 epochs with a batch size of 16 using the Adam optimizer (Kingma and Ba, 2015). The EP models were trained for three epochs, using the same batch size and optimizer. The learning rates were kept at `1e-04` for the EP tasks and `1e-05` for the fine-tuning tasks. Further details about hyper-parameter searching and the exact configurations are provided in the appendix. Following the training regime of (Pruksachatkun et al., 2020), the models were evaluated on a subset of the validation data every 500 mini-batches with early stopping on 100 evaluations.

4.2 Results

We report the Micro-F1 scores for the EP tests on fine-tuned models in Table 1 (the average over 5 runs for each experiment, and the standard deviation varies between $0.1 - 1.5\%$). See the appendix for detailed results.

For SQuAD the test data is not publicly available, therefore, we report the results on dev data.

[2]Merchant et al. (2020) and van Aken et al. (2019) reportedly used the same HPs.

	SRL	CoREF	PoS	NER
BERT-base	81.1	81.2	96.1	93.0
Fine-tuning on original data				
SQuAD (81.9)	79.9	81.2	95.3	92.4
ReCoRD (57.0)	79.7	80.9	95.8	93.4
MultiRC (63.7)	80.7	82.3	95.8	93.5
HotpotQA (77.0)	77.7	80.2	94.3	90.9
Fine-tuning on randomized data				
SQuAD (7.4)	74.8	78.9	91.7	86.8
HotpotQA (12.5)	74.0	79.5	92.0	86.2

Table 1: Micro-F1 scores for different EP tasks: without fine-tuning, with fine-tuning on the original datasets, and with fine-tuning on randomized datasets. The F1 scores for the QA datasets are given in parentheses.

We changed HotpotQA instances to SQuAD style ones by providing the relevant sentences as the context, which is given for the train and dev data, but not for the test data. Therefore, we only report the results on the dev data.

For SQuAD and MultiRC, the results are somewhat lower than the best results reported in the literature with similar architectures (88.5 for Devlin et al. (2019) and 70.4 for Wang et al. (2019)).[3] For ReCoRD, the results are slightly better (Zhang, 2020). For HotpotQA, no fair evaluation is possible due to the data modifications.

Our EP test results for SRL and CoREF do differ from the previous work (Tenney et al., 2019a), but they are comparable with Liu et al. (2019), which uses the same dataset. However, we are more concerned with the fact that the EP test results do not change significantly when a fine-tuned vs the original encoder is used. In the randomization experiments, we see that the QA F1 score drops as expected, and the EP test results do change, but not as significantly as the QA F1 scores. This also indicates that improving the performance of the QA model itself might not change the EP test results significantly.

In summary, our experiments suggest that **fine-tuning indeed does not significantly change the encoding of linguistic knowledge in the underlying encoder, *when measured by the EP tests***, which is consistent with the findings of previous work (Merchant et al., 2020; van Aken et al., 2019), but provides complementary evidence.

[3]We use a max length of 128 in the encoder, whereas a max length of 512 produces comparable results. However, the target test results and EP test results are not correlated, therefore, we do not investigate this further.

5 RQ2: EP Test for Coreference

While we can expect the model to acquire some linguistic skills (the ability to do coreference resolution, identify the part of speech of a token) by learning to perform a QA task, there is no guarantee for this: a model can reason differently than we expect it to. For example, many HotpotQA questions can be answered by identifying the necessary entity type and not the multi-hop reasoning process that we expect (Min et al., 2019).

Therefore, in $RQ2$, we want to see whether the EP test results change when we know the encoder has to acquire particular grammatical knowledge K for the QA task. Consider two models M_1 (E_1 + QA_Layer_1) and M_2 (E_2 + QA_Layer_2) in our encoder + QA layer architecture. Assume we can identify a set of questions Q that can *only* be answered using K. If M_1 performs significantly better than M_2 in these questions, we can say that E_1 has encoded more information about K than the E_2 because the QA layers are unlikely to encode that knowledge as they have much less parameters than the encoders. Therefore, in the EP test for K, we can expect E_1 to perform better than E_2.

We define M_1 as a **fine-tuned-encoder**, where the full architecture (encoder (E_1) + QA layer) is trained; and M_2 as a **frozen encoder**: the encoder (E_2) is frozen and *only* the QA layer is trained. We choose K to be the grammatical knowledge of coreference. It is difficult to understand whether the knowledge of semantic roles or part of speech would be needed to answer a question, but it possibly can be done for coreference. For example, in Figure 2a, it is relatively easy to see that to answer the question a human needs to resolve the reference 'he' in the second sentence to 'Leo Strauss'.

5.1 Finding Coreference Questions

We employed four NLP practitioners to annotate 200 questions (100 each from SQuAD and HotpotQA). Each annotator was given a sample of 100 questions and was asked to stop as soon as they found 50 positive (questions they thought required coreference) instances. The question in Figure 2a is sampled from that dataset. But the question can also be answered by a shortcut (Geirhos et al., 2020). We know a 'where' question will only be answered by an entity of type location and there are two such entities in the context: Germany and United States. Germany is an argument to the trigger verb 'born', hence, is the answer.

The Quoref dataset (Dasigi et al., 2019) reportedly consists of questions that can only be answered by understanding the concept of coreference. The annotators *design* the questions themselves, which is different from our post-hoc annotation process. Figure 2b shows a sample question from the dataset. This question can not be answered without resolving the pronominal antecedent 'he' to 'Frankie'. But even Quoref *can* have questions that can be answered with a shortcut, therefore we develop algorithms to filter them out (§5.1.1, §5.1.2).

5.1.1 Model-Agnostic Filter

In Quoref, the answer is a span in the context. The Model-Agnostic Filter algorithm works in two steps: a) **Sentence Selection**: Select the context sentence that is most similar to the question; and b) **Entity Type Matching**: The question expects an entity of a particular type, eg: 'where' $\rightarrow$ location, 'who' $\rightarrow$ person. From the sentence selected in the last step, select an entity of the same type.

For 'Sentence Selection', we use two methods: 1) **Token-Overlap**: Select the sentence that has the highest token overlap with the question tokens, and 2) **Sentence Encoder**: An off-the-shelf sentence encoder from Reimers and Gurevych (2019) trained on MS Marco (Nguyen et al., 2016), a large scale dataset for answer passage retrieval (see the appendix for details).

For the 'Entity Type Matching' step, we design both supervised and unsupervised algorithms to determine the type of the answer entity from the question. For the unsupervised algorithm, we define a map (see the appendix) with the 'wh' words (who, when, whom) as the keys and the entity type as values (who $\leftarrow$ `PER`). The first 'wh' word in the question determines the output. For example, for the question "where was Plato born, who wrote Republic?" it produces `LOC`. If no such word is found, it outputs an `UNK_ETYPE`.

This unsupervised approach will predict the wrong entity type for questions such as "Who won the World Cup in 2002?" (`PER` instead of `LOC`). Therefore, we train and evaluate supervised classification models on the training split of the Quoref dataset.[4] The label for a question is the entity type of the answer,[5] as detected by an off-the-shelf entity extractor from Spacy. If the answer is not a

[4]Further divided into 70(train)-20(dev)-10(test) splits

[5]One of the 18 types in Pradhan et al. (2013)

Context: Leo Strauss ...was a German-American political philosopher ... He was born in Germany... Thoughts on Machiavelli is a book by Leo Strauss ...
Question: Where was the author of Thoughts of Machiavelli born?
Answer: Germany

(a) A sample question from SQuAD.

Context: Frankie Bono, a mentally disturbed hitman from Cleveland,..Next, he goes to purchase a revolver from Big Ralph....
Question: What is the first name of the person who purchases a revolver?
Answer: Frankie

(b) A sample question from Quoref.

Figure 2: Sample questions from SQuAD and Quoref datasets. A reader relying on coreference resolution would take into account the green tokens.

Sentence	Etype		EM
Overlap	Supervised	fine-tuned (63)	6.31
		WordConv (58)	6.27
	Unsupervised		1.22
Encoder	Supervised	fine-tuned	5.99
		WordConv	5.48
	Unsupervised		0.97

Table 2: Different strategies for the Model-Agnostic Filter algorithm. EM stands for exact match, i.e., the percentage of cases where the filter produces the exact answer.

	frozen		fine-tuned	
	F1	EM	F1	EM
Quoref dev	10.23	5.41	69.53	65.61
- MAF	10.09	5.36	69.21	65.31
- MDF	7.00	3.19	38.57	30.85
- (MAF + MDF)	6.76	2.97	38.38	30.69
CoREF (Micro-F1)	81.68 ± 1.68		83.11 ± 0.7	

Table 3: Performance of frozen encoder and fine-tuned-encoder models when filters are applied: separately and in combination. MAF and MDF stands for model agnostic and dependent filters. The last row reports the Micro-F1 for both encoders in CoREF EP test.

named entity, or our entity extractor fails to determine its type, the label is `UNK_ETYPE`.[6] We experiment with two architectures: 1) a fine-tuned $\text{BERT}_{\text{base-cased}}$ model; and 2) a popular word convolutional model for sentence classification (Kim, 2014) using three parallel filters and 300 dimensional Google News Word2Vec representations (Mikolov et al., 2013). More details about the data, model architectures, and training is provided in the appendix.

5.1.2 Model-Dependent Filter

Following Sugawara et al. (2020) we replace all pronouns from the context in a question with random strings of the same length. If any one of M_1 or M_2 can still answer the question, it can *arguably* be answered without the knowledge of coreference.

5.1.3 Experiments & Results

The BERT and the WordConv supervised entity detectors have an average accuracy of $63.55 \pm 0.1\%$, and $58.81 \pm 0.3\%$ over 5 runs respectively.

The 'EM' column in Table 2 shows the proportion of Quoref questions (the validation split) that can be answered by the Model-Agnostic Filter algorithm. 'Overlap' and 'Encoder' are the two strategies for the 'Sentence Selection' step, and 'Supervised' and 'Unsupervised' are the same for the 'Entity Type Matching'.

The final Model-Agnostic Filter algorithm uses the token overlap approach to select a sentence from the context and uses the best fine-tuned BERT model to find the entity type for the answer. With this, we can filter out $6.3\%(155/2418)$ questions from the dev set. While this number is not very high, a similar exercise on SQuAD determines that at least 21% questions can be answered by this shortcut, which is consistent with prior findings (Jia and Liang, 2017).

In the Model-Dependent Filter algorithm, individually, the fine-tuned model filters out 55% dev instances, and the frozen encoder model filters out 6% of them, and in total, they filter out 56% (there is a large overlap).

5.2 Target Task vs Encoder EP Test

Table 3 shows the results of the fine-tuned and the frozen encoder on Quoref dev set before and after the filters are applied. As can be seen, the fine-tuned encoder performs much better than the frozen one across all scenarios. The performance of

[6]Indeed, a significant number of questions are labeled as such: 33%, 34%, and 35% in the train, dev, and test split of the data respectively.

these encoders on the CoREF EP test is given in the last column, which, however, does not differ much. This proves that **while one encoder might encode a linguistic knowledge (coreference) better than the other, the EP tests might fail to capture that**.

6 RQ3: An Analysis of EP Tests

6.1 Analysis

We see that the EP test results are surprisingly stable, even when we expect the fine-tuned encoder to learn or forget certain linguistic skills. EP tests are *indirect* measures of a representation's quality and have been criticized as such. Voita and Titov (2020) shows that for some EP tests, a large pre-trained LM (ElMo (Peters et al., 2018)) has the same performance as a random encoder. They conclude that the test measures the classifiers' ability to learn the EP task, and not the knowledge encoded in the representations itself. They propose using an information-theoretic (minimum description length or MDL) probe.

In a similar vein, Hewitt and Liang (2019) suggests designing a control task. A control task for an EP test is the same classification task, only the labels of the original task are changed so that it can not be recovered from the linguistic information. For example, two tokens with different part-of-speech tags will be mapped to the same arbitrary label. A classifier (probe) that performs well on both the control task and the original EP test must be learning the correlations in the data, and not using any information from the representations. Therefore, it can not measure the encoding of grammatical knowledge in the encoders.

If an EP test has only two labels, they will just be flipped in the control task. This makes the control task and the original EP test the same problem for the probe, and they must have the same performance. Therefore, for binary EP tests such as CoREF (the one we are interested in), no control task can be designed, but we use simple linear probes following Hewitt and Liang (2019)'s recommendations.

Had we used MDL probes instead, would our conclusions in $RQ1$ and $RQ2$ change? Prior work reports that the fine-tuned encoders do not show much difference in the MDL probes themselves (Merchant et al., 2020). Moreover, even in the original MDL probe paper, the CoREF test results are similar in a pre-trained vs random encoder (Voita and Titov, 2020). Therefore, replacing the current EP tests with MDL probes should not change the findings for the previous research questions.

	SRL	CoREF	PoS	NER
mem_uniform	32.46	65.02	88.62	71.59
mem_freq	44.45	78.06	88.69	73.27
same_prec_ante	-	70.23%	-	-
BERT-base	81.08	81.2	96.11	93.06

Table 4: A performance comparison on EP test results: Micro-F1 scores for heuristics and $\text{BERT}_{\text{base-uncased}}$ models (average over 5 runs, STD. varies from 0.09 – 1%).

Does this conclusively mean that the linguistic skill is not improved in the encoder, even when the task calls for it? Both Merchant et al. (2020) and van Aken et al. (2019) arrive at that conclusion, albeit without fine-tuning on a skill-specific target dataset such as Quoref. We present a dataset bias explanation (§6.2). Note the EP test dataset is used in both Merchant et al. (2020), van Aken et al. (2019), therefore the same explanation is valid for both these studies.

6.2 EP Test Heuristics

Following Gururangan et al. (2018), we design unsupervised algorithms that exploits spurious correlations in the dataset.

- **Memorization**: If a test data point is in the training data, the classifier returns the training data label. Else, it returns a random label either a) uniformly sampled ('mem_uniform') or b) sampled from the label probability distribution of the training data ('mem_freq').
- **Same Precedent-Antecedent**: In the CoREF dataset, whenever the precedent and antecedent are the same ("Obama is the president of the US. *He* lives in Washington D.C. *He* went to Harvard."), return positive.

6.2.1 Results

Table 4 shows the results for various heuristics and a $\text{BERT}_{\text{base-uncased}}$ encoder. To achieve moderately high performance on most EP tests, no specific representation is needed, let alone from a pre-trained or a fine-tuned one. The *Same Precedent-Antecedent* heuristic has a Micro-F1 score of 70.23% when used alone. This can be combined with mem_freq/ mem_uniform, but the combination provides no significant improvement. Overall, mem_freq is a strong baseline for the EP tests.

		Best	Worst
SRL	overall	-	-7.05
	easy	-	-4.40
	hard	-	-8.47
CoREF	overall	+1.93	-2.32
	easy	**+4.51**	**-7.11**
	hard	+0.47	-0.17
PoS	overall	-	-4.12
	easy	-	-4.01
	hard	-	**-12.30**
NER	overall	+0.35	-6.85
	easy	+0.25	-4.82
	hard	+0.62	**-12.48**
CoREF-LS	overall	+1.93	-2.32
	easy	-1.22	+1.1
	hard	**+13.27**	**-14.68**

Table 5: The accuracy changes across easy and hard instances for the best and worst fine-tuned models. For SRL and PoS, BERT-base is the best, therefore, there is no positive change. CoREF-LS refers to the case when the easy and hard points are created by splitting the data across labels. In some splits, the changes are significantly different than the overall change.

6.3 EP Tests: Hard & Easy Instances

The results in §6.2 can be viewed in another way. Many instances in the EP test data can be solved by memorization, i.e., are *easy* instances. But there are definitely difficult ones that account for the performance difference in the heuristics and the $\text{BERT}_{\text{base-uncased}}$ model. When the fine-tuned encoders show marginal improvements or deterioration (over the base encoders) do the performance in these EP tests increase/decrease uniformly across the hard/easy instances, or in the harder instances the results change more drastically? **In the second case, it can be argued that the EP tests do actually capture the change in the encoders, but the change is artificially clamped, which is an unfortunate side-effect of the dataset**.

We first divide the test data for an EP test into easy and hard instances: the ones that can be solved by mem_freq or not. Then we note the average accuracy of base $\text{BERT}_{\text{base-uncased}}$ encoders on these splits. Finally, we take two QA models (from two separate datasets) for which the encoders had the average best/worst results in the said EP test. Do the results change from the base encoder similarly across these splits?

6.3.1 Results

Table 5 shows no discerning pattern in the 'Best fine-tuned' column, probably because the overall improvements are indeed not significant. However, when an encoder model performs poorly (the 'Worst fine-tuned' column), it performs disproportionately badly on the hard instances. This proves that while on the surface it might appear the results are similar (Merchant et al., 2020; van Aken et al., 2019), they are indeed not.

We are however more interested in CoREF, because as discussed in $RQ2$, we have reasons to believe that some encoders have indeed learnt more about the skill of coreference than the others. However, contrary to the other results, it seems that the better/worse results come from the easy instances.

CoREF dataset has a significant label imbalance (compare row 1 and 2 in Table 4). A classifier predicting a negative label for all test instances can achieve an accuracy of 78.33%. If we split the data across the labels (CoREF-LS), we see that the results change drastically, with a clear indication that a better encoder gets better by classifying the hard (positive) instances better, and a worse encoder fails harder on the same instances. Unsurprisingly, the better encoders come from the encoders trained on the Quoref dataset.

In summary, we show *why* an EP model can fit well to the EP data without using a *good* representation. This indicates that while fine-tuning may improve the encoding of grammatical knowledge in encoders, the current EP tests (even the MDL probes) might not be able to capture it. **There are issues with the datasets rather than the task design itself**. This is a new explanation for the apparent consistency of EP test results in fine-tuned models, whereas previous work has mostly focused on classifier knowledge (see §6.1).

7 Conclusion and Future Work

Edge probing tests are the predominant method to probe for linguistic information in large language models. We use them to evaluate how the process of fine-tuning an LM for QA might change the grammatical knowledge in an encoder, and observe no significant differences between pre-trained and fine-tuned LMs. More importantly, we find this phenomenon in carefully designed target tasks where the models must use the said grammatical knowledge. From similar EP test results, previous works have concluded that fine-tuning does not change the encoding of grammatical knowledge. However, our analysis provides a 'dataset bias' explanation for the consistency of the results and provides some clues as to why *any* representation tends to achieve very similar results for EP tests. This is different from the previous task-design criticisms of the EP tests.

Do fine-tuned NLU models score highly on benchmarks for the right reasons, i.e., follow the human reasoning process? This work shows some evidence in favor of that. The encoding of grammatical knowledge in QA encoders is improved as expected when the models are trained on the right datasets, and the dataset biases in the EP tests are corrected. In light of this evidence, in the future, we plan to identify the exact reasoning steps in the QA models through post-hoc explainability methods and study whether they align with the human reasoning steps.

8 Acknowledgement

We thank Behzad Golshan and Wang-Chiew Tan for the helpful discussions and annotation work. This project has been partly funded by a Megagon Labs research grant.

References

Yonatan Belinkov. 2022. Probing Classifiers: Promises, Shortcomings, and Advances. *Computational Linguistics*, 48(1):207–219.

Yonatan Belinkov and James R. Glass. 2019. Analysis Methods in Neural Language Processing: A Survey. *Trans. Assoc. Comput. Linguistics*, 7:49–72.

Terra Blevins, Omer Levy, and Luke Zettlemoyer. 2018. Deep RNNs Encode Soft Hierarchical Syntax. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 2: Short Papers*, pages 14–19. Association for Computational Linguistics.

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D. Manning. 2019. What Does BERT Look at? An Analysis of BERT's Attention. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP, BlackboxNLP@ACL 2019, Florence, Italy, August 1, 2019*, pages 276–286. Association for Computational Linguistics.

Pradeep Dasigi, Nelson F. Liu, Ana Marasovic, Noah A. Smith, and Matt Gardner. 2019. Quoref: A Reading Comprehension Dataset with Questions Requiring Coreferential Reasoning. 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 2019, Hong Kong, China, November 3-7, 2019*, pages 5924–5931. Association for Computational Linguistics.

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, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Allyson Ettinger, Ahmed Elgohary, and Philip Resnik. 2016. Probing for semantic evidence of composition by means of simple classification tasks. In *Proceedings of the 1st Workshop on Evaluating Vector-Space Representations for NLP, RepEval@ACL 2016, Berlin, Germany, August 2016*, pages 134–139. Association for Computational Linguistics.

Robert Geirhos, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard Zemel, Wieland Brendel, Matthias Bethge, and Felix A. Wichmann. 2020. Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2(11):665–673.

José Luis Vicedo González and Antonio Ferrández Rodríguez. 2000. Importance of Pronominal Anaphora Resolution in Question Answering Systems. In *38th Annual Meeting of the Association for Computational Linguistics, Hong Kong, China, October 1-8, 2000*, pages 555–562. ACL.

Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel R. Bowman, and Noah A. Smith. 2018. Annotation Artifacts in Natural Language Inference Data. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 2 (Short Papers)*, pages 107–112. Association for Computational Linguistics.

Karl Moritz Hermann, Tomás Kociský, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching Machines to Read and Comprehend. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 1693–1701.

John Hewitt and Percy Liang. 2019. Designing and Interpreting Probes with Control Tasks. 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 2019, Hong Kong, China, November 3-7, 2019*, pages 2733–2743. Association for Computational Linguistics.

Robin Jia and Percy Liang. 2017. Adversarial Examples for Evaluating Reading Comprehension Systems. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*, pages 2021–2031. Association for Computational Linguistics.

Divyansh Kaushik and Zachary C. Lipton. 2018. How much reading does reading comprehension require? A critical investigation of popular benchmarks. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 5010–5015. Association for Computational Linguistics.

Daniel Khashabi, Snigdha Chaturvedi, Michael Roth, Shyam Upadhyay, and Dan Roth. 2018. Looking Beyond the Surface: A Challenge Set for Reading Comprehension over Multiple Sentences. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 252–262, New Orleans, Louisiana. Association for Computational Linguistics.

Yoon Kim. 2014. Convolutional Neural Networks for Sentence Classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1746–1751. ACL.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Olga Kovaleva, Alexey Romanov, Anna Rogers, and Anna Rumshisky. 2019. Revealing the Dark Secrets of BERT. 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 4365–4374, Hong Kong, China. Association for Computational Linguistics.

Kenton Lee, Luheng He, Mike Lewis, and Luke Zettlemoyer. 2017. End-to-end Neural Coreference Resolution. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*, pages 188–197. Association for Computational Linguistics.

Tal Linzen, Emmanuel Dupoux, and Yoav Goldberg. 2016. Assessing the ability of lstms to learn syntax-sensitive dependencies. *Trans. Assoc. Comput. Linguistics*, 4:521–535.

Nelson F. Liu, Matt Gardner, Yonatan Belinkov, Matthew E. Peters, and Noah A. Smith. 2019. Linguistic Knowledge and Transferability of Contextual Representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 1073–1094. Association for Computational Linguistics.

Amil Merchant, Elahe Rahimtoroghi, Ellie Pavlick, and Ian Tenney. 2020. What Happens To BERT Embeddings During Fine-tuning? In *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP, BlackboxNLP@EMNLP 2020, Online, November 2020*, pages 33–44. Association for Computational Linguistics.

Tomás Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space. In *1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings*.

Sewon Min, Eric Wallace, Sameer Singh, Matt Gardner, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2019. Compositional Questions Do Not Necessitate Multi-hop Reasoning. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 4249–4257. Association for Computational Linguistics.

Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. 2016. MS MARCO: A Human Generated MAchine Reading COmprehension Dataset. In *Proceedings of the Workshop on Cognitive Computation: Integrating neural and symbolic approaches 2016 co-located with the 30th Annual Conference on Neural Information Processing Systems (NIPS 2016), Barcelona, Spain, December 9, 2016*, volume 1773 of *CEUR Workshop Proceedings*. CEUR-WS.org.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep Contextualized Word Representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Hwee Tou Ng, Anders Björkelund, Olga Uryupina, Yuchen Zhang, and Zhi Zhong. 2013. Towards Robust Linguistic Analysis using OntoNotes. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning, CoNLL 2013, Sofia, Bulgaria, August 8-9, 2013*, pages 143–152. ACL.

Yada Pruksachatkun, Phil Yeres, Haokun Liu, Jason Phang, Phu Mon Htut, Alex Wang, Ian Tenney, and Samuel R. Bowman. 2020. jiant: A Software Toolkit for Research on General-Purpose Text Understanding Models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 109–117, Online. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100, 000+ Questions for Machine Comprehension of Text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, pages 2383–2392. The Association for Computational Linguistics.

Sagnik Ray Choudhury, Anna Rogers, and Isabelle Augenstein. 2022. Machine Reading, Fast and Slow: When Do Models "Understand" Language? In *Proceedings of the 29th International Conference on Computational Linguistics*. International Committee on Computational Linguistics.

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.

Anna Rogers, Matt Gardner, and Isabelle Augenstein. 2022. QA Dataset Explosion: A Taxonomy of NLP Resources for Question Answering and Reading Comprehension. *Computing Surveys (CSUR)*, to appear.

Priyanka Sen and Amir Saffari. 2020. What do models learn from question answering datasets? In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 2429–2438. Association for Computational Linguistics.

Dan Shen and Mirella Lapata. 2007. Using Semantic Roles to Improve Question Answering. In *EMNLP-CoNLL 2007, Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, June 28-30, 2007, Prague, Czech Republic*, pages 12–21. ACL.

Xing Shi, Inkit Padhi, and Kevin Knight. 2016. Does String-Based Neural MT Learn Source Syntax? In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, pages 1526–1534. The Association for Computational Linguistics.

Saku Sugawara, Pontus Stenetorp, Kentaro Inui, and Akiko Aizawa. 2020. Assessing the Benchmarking Capacity of Machine Reading Comprehension Datasets. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 8918–8927. AAAI Press.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019a. BERT Rediscovers the Classical NLP Pipeline. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 4593–4601. Association for Computational Linguistics.

Ian Tenney, Patrick Xia, Berlin Chen, Alex Wang, Adam Poliak, R. Thomas McCoy, Najoung Kim, Benjamin Van Durme, Samuel R. Bowman, Dipanjan Das, and Ellie Pavlick. 2019b. What do you learn from context? probing for sentence structure in contextualized word representations. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Ke M. Tran, Arianna Bisazza, and Christof Monz. 2018. The importance of Being Recurrent for Modeling Hierarchical Structure. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 4731–4736. Association for Computational Linguistics.

Betty van Aken, Benjamin Winter, Alexander Löser, and Felix A. Gers. 2019. How Does BERT Answer Questions?: A Layer-Wise Analysis of Transformer Representations. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management, CIKM 2019, Beijing, China, November 3-7, 2019*, pages 1823–1832. ACM.

Elena Voita and Ivan Titov. 2020. Information-Theoretic Probing with Minimum Description Length. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 183–196. Association for Computational Linguistics.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. SuperGLUE: A Stickier Benchmark for General-Purpose Language Understanding Systems. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 3261–3275.

Ralph Weischedel, Martha Palmer, Mitchell Marcus, Eduard Hovy, Sameer Pradhan, Lance Ramshaw, Nianwen Xue, Ann Taylor, Jeff Kaufman, Michelle Franchini, et al. 2013. Ontonotes release 5.0 ldc2013t19. *Linguistic Data Consortium, Philadelphia, PA*, 23.

Jason Weston, Antoine Bordes, Sumit Chopra, and Tomas Mikolov. 2016. Towards AI-Complete Question Answering: A Set of Prerequisite Toy Tasks. In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A Dataset for Diverse, Explainable Multi-hop Question Answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 2369–2380. Association for Computational Linguistics.

Matthew D. Zeiler. 2012. ADADELTA: An Adaptive Learning Rate Method. *CoRR*, abs/1212.5701.

Sheng Zhang. 2020. BERT fine-tuning re-implementation for ReCoRD, JHU.

Sheng Zhang, Xiaodong Liu, Jingjing Liu, Jianfeng Gao, Kevin Duh, and Benjamin Van Durme. 2018. ReCoRD: Bridging the Gap between Human and Machine Commonsense Reading Comprehension. *CoRR*, abs/1503.06733.

A Appendix A: RQ1

A.1 QA Datasets and Model Architectures

SQuAD: We use the first (`V1.1`) version of the SQuAD dataset, which contains around 100K question, answer, context triples collected from the English Wikipedia. The answers are spans within the context. The questions and contexts are concatenated, and a linear layer on top of a contextual encoder is used to predict the probability of a context token i being the start ($P_{i,s}$) or end ($P_{i,e}$) of an answer. The score ($S_{i,j}$) for a span with start token i and end token j is $P_{i,s} + P_{j,e}$. For all valid combinations of i and j, the span with the highest score is chosen as the answer. A cross-entropy loss between the actual and predicted start/end positions is minimized.

HotpotQA: The HotpotQA dataset is a collection of 113K question-answer pairs, with two improvements over SQuAD: 1) Each context consists of multiple paragraphs (as opposed to a single one) and the model needs to reason over some of them to provide an answer and 2) The sentences required to answer a question are provided (as paragraph index, sentence index). The dataset features an implicit IR task (finding the relevant sentences), therefore we reduce it to a SQuAD style one by changing the context to the full paragraphs from where the supporting facts are chosen and removing the questions where the answer spans cannot be found in the context (3.8 % in train and 3.9 % in dev data).

ReCoRD: The ReCoRD dataset contains a set of 120,000 Close style questions from the CNN/Daily Mail dataset (Hermann et al., 2015). A Cloze style query is a statement with an occluded entity that is factually supported by a passage. The dataset provides the named entities in the context, one of which is the answer. For each (question, context, N named entity) triple in the data, N (question, context, label) triples are created, with the missing entity in the question replaced by one of the provided entities, and the label is put as true or false depending on whether the entity is the answer or not. This NLI style formulation reduces the QA task to a classification problem, for which a two-layer MLP is used on top of the encoder layer: a linear layer with a tanh activation, followed by another linear layer.

MultiRC: MultiRC is a multiple-choice QA dataset where the model has to choose one or multiple of provided answers utilizing the text from the question, context, and the answer itself. We reduce it to a binary classification task, where the input is a concatenation of the question, the context, and the answer (for each of the possible answers). The same architecture as ReCoRD is used.

A.2 HP Searching and Final Configurations for QA Datasets and EP Tests

Training Details: We searched for the following hyper-parameters (HPs): number of epochs and learning rates. Finally, the QA models were trained for 10 epochs with a batch size of 16 using the Adam optimizer (Kingma and Ba, 2015). The EP models were trained for the three epochs, using the same batch size and optimizer. The learning rates were kept at `1e-04` for the EP tasks and `1e-05` for the fine-tuning tasks. Following the training regime of (Pruksachatkun et al., 2020), the model was evaluated on a subset of the validation data every 500 mini-batches with early stopping on 100 evaluations. The config files in the provided code show the detailed HPs. For the EP tests, the hyper-parameters are all same as the baseline (Tenney et al., 2019b) except for the batch size (32 theirs vs 16 ours). Merchant et al. (2020) and van Aken et al. (2019) reportedly used the same HPs.

A.3 Reproducibility Checklist

Description of computing infrastructure used: Titan RTX GPU, CUDA version 11.2.

The average runtime for each model or algorithm (e.g., training, inference, etc.), or estimated energy cost: 5-6 Hours for EP tests, 3-4 hours for QA models.

Number of parameters in each model: For QA models: BERT base uncased parameters + 128*num_classes (FC layer). For EP tests, they are the same.

Corresponding validation performance for each reported test result: For three QA datasets, we use the validation data itself. For the EP models, validation results are very similar to the test results reported.

Explanation of evaluation metrics used, with links to code: The evaluation metric for QA models is F1, as common in most QA datasets, including SQuAD. For EP tests, the evaluation metric is Micro-F1, which is used in Tenney et al. (2019b), the paper which is our baseline. The implementations are from Pruksachatkun et al. (2020), which hosts the original code used in Tenney et al. (2019b). We also use

For all experiments with hyperparameter search:

model	acc	micro_f1	macro_f1	weighted _f1
BERT-base	81.08 ± 1.33	81.08 ± 1.33	31.93 ± 1.34	80.09 ± 1.54
fine-tuned on original data				
SQuAD	79.97 ± 1.1	79.97 ± 1.1	30.57 ± 1.0	78.86 ± 1.3
ReCoRD	79.71 ± 1.46	79.71 ± 1.46	30.47 ± 1.61	78.61 ± 1.7
MultiRC	80.69 ± 1.25	80.69 ± 1.25	31.76 ± 1.37	79.68 ± 1.47
Hotpot	77.73 ± 1.2	77.73 ± 1.2	28.4 ± 1.23	76.45 ± 1.43
finetuned on randomized data				
SQuAD	74.79 ± 0.15	74.79 ± 0.15	26.07 ± 0.56	73.36 ± 0.17
Hotpot	74.03 ± 0.33	74.03 ± 0.33	25.79 ± 0.38	72.5 ± 0.35

Table 6: Results for SRL EP test.

The exact number of training and evaluation runs: For each QA model, 5 training/evaluation runs. For each EP test, 5 training/eval run.
Bounds for each hyperparameter: training batch size: 8-32, learning rate: for QA models, `1e-05` to `1e-03`, 3 steps. For EP tests, `1e-06` to `1e-04`, 5 steps.
Hyperparameter configurations for best-performing models: Provided as config files.
The method of choosing hyperparameter values (e.g., uniform sampling, manual tuning, etc.) and the criterion used to select among them (e.g., accuracy): Uniform sampling, F1 for QA models in dev data, Micro-F1 for EP tests in test data.

A.4 Results

Detailed results for the experiments in §4 are provided below in Tables 6, 7, 8, and 9.

B Appendix B: RQ2

B.1 Sentence Embedding Model for Answer Sentence Selection

MS MARCO (Nguyen et al., 2016) is a large dataset of question/ answer pairs. The dataset was built by first sampling queries from Bing search log and then using Bing to retrieve relevant documents and automatically extract relevant passages from those documents. Annotators were asked to mark relevant spans from the documents as answers.

We us a sentence embedding model (Reimers and Gurevych, 2019) built on the MS MARCO

model	acc	micro_f1	macro_f1	weighted _f1
BERT-base	81.18 ± 1.68	81.18 ± 1.68	59.33 ± 7.99	76.2 ± 3.89
fine-tuned on original data				
Quoref	83.11 ± 0.7	83.11 ± 0.7	67.97 ± 2.05	80.45 ± 1.09
SQuAD	81.19 ± 1.49	81.19 ± 1.49	60.27 ± 8.48	76.58 ± 4.01
ReCoRD	80.88 ± 1.76	80.88 ± 1.76	58.08 ± 9.79	75.57 ± 4.66
MultiRC	82.37 ± 1.71	82.37 ± 1.71	65.12 ± 8.7	78.99 ± 4.19
Hotpot	80.19 ± 1.84	80.19 ± 1.84	55.38 ± 8.86	74.21 ± 4.31
finetuned on randomized data				
SQuAD	78.86 ± 0.38	78.86 ± 0.38	48.77 ± 3.29	71.01 ± 1.5
Hotpot	79.45 ± 0.23	79.45 ± 0.23	52.1 ± 2.31	72.61 ± 1.03

Table 7: Results for CoREF EP test

model	acc	micro_f1	macro_f1	weighted _f1
BERT-base	96.11 ± 0.15	96.11 ± 0.15	87.88 ± 0.8	96.06 ± 0.15
fine-tuned on original data				
SQuAD	95.27 ± 0.04	95.27 ± 0.04	87.08 ± 0.45	95.2 ± 0.05
ReCoRD	95.77 ± 0.08	95.77 ± 0.08	86.91 ± 0.39	95.71 ± 0.09
MultiRC	95.77 ± 0.19	95.77 ± 0.19	86.51 ± 1.29	95.71 ± 0.19
Hotpot	94.31 ± 0.09	94.31 ± 0.09	83.45 ± 0.7	94.21 ± 0.1
fine-tuned on randomized data				
SQuAD	91.74 ± 0.35	91.74 ± 0.35	78.52 ± 1.15	91.54 ± 0.37
Hotpot	91.99 ± 0.13	91.99 ± 0.13	79.59 ± 0.65	91.8 ± 0.14

Table 8: Results for PoS EP test

model	acc	micro_f1	macro_f1	weighted _f1
BERT-base	93.0 ± 0.28	93.0 ± 0.28	78.12 ± 1.1	92.29 ± 0.37
fine-tuning on original data				
SQuAD	92.44 ± 0.09	92.44 ± 0.09	78.19 ± 0.47	91.86 ± 0.12
ReCoRD	93.35 ± 0.24	93.35 ± 0.24	79.88 ± 0.8	92.79 ± 0.31
MultiRC	93.5 ± 0.4	93.5 ± 0.4	80.51 ± 1.54	93.0 ± 0.52
Hotpot	90.9 ± 0.16	90.9 ± 0.16	73.81 ± 0.62	89.94 ± 0.21
fine-tuning on randomized data				
SQuAD	86.77 ± 1.02	86.77 ± 1.02	64.05 ± 3.64	85.25 ± 1.26
Hotpot	86.15 ± 0.32	86.15 ± 0.32	61.56 ± 1.42	84.48 ± 0.37

Table 9: Results for NER EP test

dataset. The pre-trained model is available off-the-shelf,[7] therefore can be used directly to find the most 'similar' sentence to the question. Among many sentence embedding models available for semantic search, we use this one because it is specifically trained on a question-passage retrieval dataset using a bi-encoder model. During training, the questions and the relevant/non-relevant passages are passed through a contextual encoder and their pooled representations are compared. The model is trained with the following objective: the (query, positive_passage) pair is supposed to be close in the vector space, while (query, negative_passage) should be distant.

B.2 Unsupervised Entity Type Selection

In unsupervised entity type selection method, we use a map to determine the entity type of the answer for a question. The map is given below:

```
   {
'how far':  [QUANTITY],
'how long':  [DATE],
'how many':  [CARDINAL],
'how old':  [QUANTITY],
'what':  [PRODUCT, WORK_OF_ART],
'when':  [DATE, TIME],
'where':  [FAC, LOC, ORG, GPE],
'who':  [PERSON],
'whom':  [PERSON],
'whose':  [PERSON, ORG, NORP]
}
```

The entity types are defined in Pradhan et al. (2013). If the question has the phrase 'how far', the returned entity type is QUANTITY. The map is an `OrderedDict`, i.e., if the question is 'how old is the person who wrote Harry Potter', the returned entity type is QUANTITY. When there are multiple possibilities ('where'), one is returned randomly.

B.3 Supervised Entity Type Selection

B.3.1 Dataset

The dataset is created using the training set of Quoref which is divided into train/dev/test split for entity type detector model training and evaluation. A sample data point is shown in Figure 3.

The data distribution is shown in Figure 4. As can be seen, it is very skewed.

[7]https://www.sbert.net/docs/pretrained-models/msmarco-v5.html

Text: What is the full name of the person who is the television reporter that brings in a priest versed in Catholic exorcism rites?
Label: PER

Figure 3: A sample instance for answer entity type classifier.

	acc	macro_f1
BERT$_{\text{base-cased}}$	63.55 ± 0.00	19.65 ± 0.04
WordConv	58.81 ± 0.01	13.31 ± 0.02

Table 10: Results for supervised entity type selection

B.3.2 Models

We use two types of models: 1) a fine-tuned 12 layer 768 dimensional BERT$_{\text{base-cased}}$ model; and 2) a popular word convolutional model for sentence classification (Kim, 2014) using three parallel filters (size 3, 4, and 5) and 300 dimensional Google News Word2Vec representations (Mikolov et al., 2013).

BERT model: This model is trained for 5 epochs, with Adam optimizer (Kingma and Ba, 2015) with a weight decay of `1.0e-08` and a learning rate of `1.0e-05`. The sequence max length is kept at 128. We search for two hyper-parameters: 1) number of epochs: 3-7, increasing by 1; and 2) learning rate: `1.0e-05`, `5.0e-05`, `1.0e-04`. Accuracy is used as the early stopping metric.

WordConv model: This model is trained for 40 epochs, with Adadelta optimizer (Zeiler, 2012) with a learning rate of `1.0e-05`. The sequence max length is again kept at 128. Accuracy is used as the early stopping metric.

B.4 Results

The results are provided in Table 10.

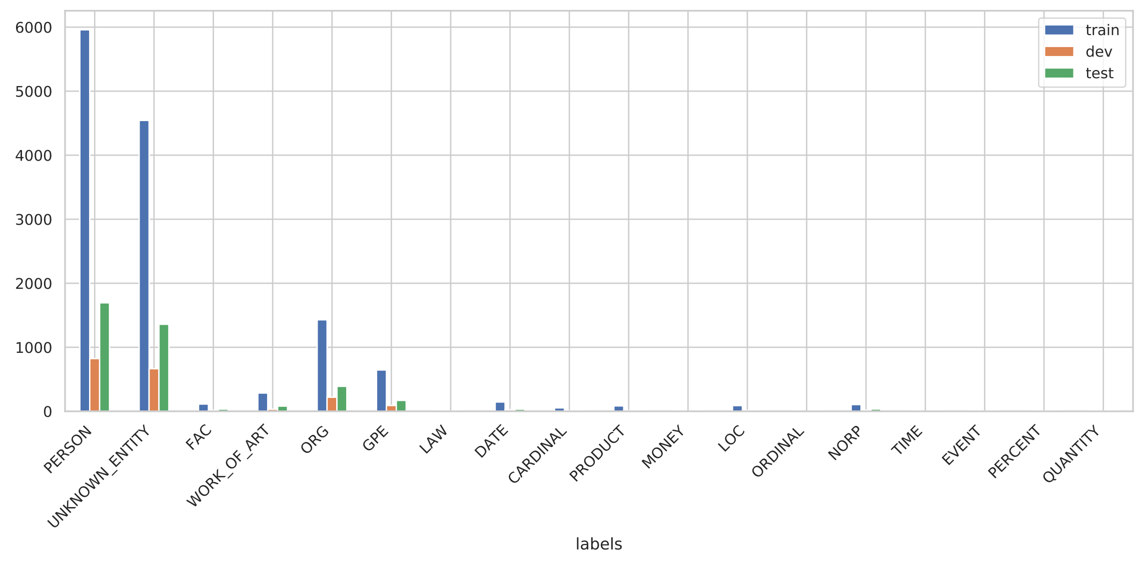

Figure 4: Label distribution for supervised entity type.