

Trained on 100 million words and still in shape: BERT meets British National Corpus

David Samuel, Andrey Kutuzov, Lilja Øvrelid and Erik Velldal

University of Oslo, Language Technology Group

{davisamu, andreku, liljao, erikve}@ifi.uio.no

Abstract

While modern masked language models (LMs) are trained on ever larger corpora, we here explore the effects of down-scaling training to a modestly-sized but representative, well-balanced, and publicly available English text source – the British National Corpus. We show that pre-training on this carefully curated corpus can reach better performance than the original BERT model. We argue that this type of corpora has great potential as a language modeling benchmark. To showcase this potential, we present fair, reproducible and data-efficient comparative studies of LMs, in which we evaluate several training objectives and model architectures and replicate previous empirical results in a systematic way. We propose an optimized LM architecture called LTG-BERT.

1 Introduction

In the pursuit of state-of-the-art performance, NLP practitioners utilize increasingly larger amounts of data to pre-train language models, making it difficult to disentangle the improvements made by the proposed modeling choices themselves. Instead, our aim is to shift the focus towards more efficient language modeling on a small and standardizable pre-training corpus. We study the data efficiency of current language models on an openly available corpus of approximately 100M words – incidentally the estimated amount of words processed by humans before adulthood (Linzen, 2020).

The goal of this paper is not to rival the paradigm of ‘massively pre-trained language models’; instead we would in this work like to pursue a complementary direction of language modeling, which will hopefully lead to more interest in data-efficient language models. In particular, our contribution in this paper is twofold – we show that:

1. 100M words is enough to train a competitive language model that outperforms the downstream performance of the original BERT model. We show

that the combination of a well-curated representative corpus, improved LTG-BERT architecture and a better training objective results in a model with stronger linguistic knowledge than the original English BERT pre-trained on 30× larger corpus.

Large language models are notoriously data hungry, requiring hundreds of gigabytes of raw textual data. This becomes a major obstacle for low-resource languages while also putting a limit to the efficiency of any ‘efficient’ language model. On top of that, the size of web-crawled corpora makes it almost impossible to control their content and to prevent learning from harmful or copyrighted text (Bender et al., 2021). The British National Corpus (BNC; Consortium, 2007) is a 100-million-word reference corpus, manually curated to cover most aspects of 20th century British English.

2. Reproducibility and fair comparison of language models can be easily achieved by pre-training on the British National Corpus.

Massive language models are often pre-trained on nonpublic filtered collections of web-crawled text, which makes any reproduction impossible. We pre-train our models on a small and publicly available corpus, which allows for a replicable comparison of different language modeling configurations and which can be easily utilized in future research of novel variants of language models. We also release the pre-processing scripts, training scripts as well as the final model checkpoints.¹

Previously, language models have been pre-trained on different corpora tokenized by different tokenizers and fine-tuned by increasingly complex learning methods. This makes any comparison of the underlying neural architectures and pre-training objectives unfair. We make the language models in this paper directly comparable by fixing the training corpus, the tokenizer and the evaluation methods, while keeping them as simple as possible.

¹<https://github.com/ltgoslo/ltg-bert>

2 Related Work

The data requirements of language models have been growing in orders of magnitude since their early stages (Jelinek, 1976). Taking a huge leap towards more recent work, ELMo (Embeddings from Language Models; Peters et al., 2018) were the first to introduce deep *contextualized* embeddings of words. Recognizing the need of a large text corpus for this task, ELMo was trained on the 1B Word Benchmark (Chelba et al., 2014). Later, BERT (Bidirectional Encoder Representations from Transformers; Devlin et al., 2019) further advanced the performance of contextualized embeddings when it based the entire language model on the Transformer architecture (Vaswani et al., 2017). Another important aspect of BERT is that it was trained on a larger corpus than ELMo: about 3.3B words from crawled English Wikipedia and BookCorpus (Zhu et al., 2015). To our best knowledge, the exact version of neither of the two subcorpora is publicly available.² The issue of limited replicability has become even more pronounced with later large language models: XLNet (Yang et al., 2019) was trained on 33B words, RoBERTa (Liu et al., 2019) on more than 30B words and GPT-3 (Brown et al., 2020) on an approximately 400B word corpus. None of these datasets is available; the authors utilize non-trivial filtering algorithms on openly available web crawls but do not release the end product nor the filtering source code.

The effect of corpus size has been thoroughly studied in Zhang et al. (2021) as well as in Hoffmann et al. (2022). They test differently sized random subsets of a BERT-like corpus (crawled Wikipedia and Smashwords) and of a massive web-crawled text corpus (MassiveText; Rae et al., 2021), respectively. Unlike them, we evaluate the effect of training on a small corpus, which was *carefully curated* to create a representative sample of English. The British National Corpus is arguably more diverse and informative than a random subset of a web crawl – hence we test how the *quality* of a pre-training corpus influences the downstream performance, not only how the data quantity matters. We believe this aspect is vital for the future research of effective and reliable language models.

²BookCorpus (Zhu et al., 2015) is not available anymore and the authors of BERT do not specify what version of Wikipedia dump they used or how did they preprocess it (<https://github.com/google-research/bert#pre-training-data>).

| | documents | sentences | words | subwords |
|-------------|-----------|-----------|-------------|-------------|
| train | 4 014 | 8 501 376 | 115 870 549 | 131 392 103 |
| development | 35 | 106 566 | 1 215 306 | 1 367 570 |

Table 1: Size of the train-development splits for the pre-processed BNC corpus. Note that the number of words is larger than the 100 million reported by the BNC Consortium due to our less conservative pre-tokenization strategy.

3 British National Corpus

We use the British National Corpus (BNC) as a diverse, balanced, compact, and publicly available monolingual English corpus. BNC is comprised of both written and spoken language with a total of 100 million words. The manually curated content contains a wide range of British English from the late 20th century – newspapers, journals, books (academic and fiction), letters, essays, unscripted informal conversations or transcribed business meetings, radio shows or phone calls. The written part makes up approximately 90% of the corpus and the remaining 10% contains the transcribed speech. The sources are truncated to contain at most 45 000 words to ensure greater diversity within the limited amount of 100 million words.

Creation. The process of creating the BNC is extensively described in its documentation on the website.³ It was created by the so called ‘BNC Consortium’ led by Oxford University Press, and including major dictionary publishers Longman and Larousse Kingfisher Chambers; academic research centres at Oxford University Computing Services, the University Centre for Computer Corpus Research on Language (UCREL) at Lancaster University, and the British Library’s Research and Innovation Centre. The purpose of the British National Corpus project was to construct a balanced and representative sample of current British English at the time. It was created over a period of four years and was a result of careful planning and data selection across a number of selection criteria (domain, time, medium, level) with proportions in the corpus designed to reflect the proportions found in real language use. It is widely acknowledged that the BNC has been a major influence on the construction of language corpora (Burnard, 2002). One downside of the BNC is that it does not re-

³<https://ota.bodleian.ox.ac.uk/repository/xmlui/handle/20.500.12024/2554>

flect anything occurring to English language and the world in the 21st century, but still no better alternatives of the same size and quality exists. In addition, BNC was used as a model for creating representative corpora for other languages: e.g., Turkish (Aksan et al., 2012).

Version. We use the third release of the corpus, BNC XML Edition (2007), which is the final revision of the texts compiled from 1991 to 1994 (Consortium, 2007). The XML edition did not get any additional content on top of the original text samples, but it got some minor corrections, more metadata and it is supplied in a convenient XML format.

3.1 Preprocessing

We convert the XML version of BNC into the Markdown format,⁴ to make it human-readable and usable as a direct raw-text input of a language model. On top of that, it can also preserve some meta-information encoded in the original XML format. Short samples from the preprocessed corpus can be found in Appendix A. After preprocessing, the articles are randomly placed into a training split and a development split. The proportions of both splits are given in Table 1.

Composition. BNC is hierarchically composed of the following text units: words, sentences, paragraphs and articles. We preserve the sentence information by storing each sentence on a separate line; paragraphs are divided by a blank line and an article always starts with a top-level header. The word-tokens are intentionally not preserved – instead, we heuristically detokenize the text to move it towards the natural text distribution. BNC includes information about the original whitespace, but we found it unreliable in some cases, necessitating the use of heuristics.

Other metadata. Other meta information available in our Markdown version is as follows:

1. **Headers:** We keep the headers together with their level by converting them to the atx-style format prefixed by hash symbols ‘#’.
2. **Speakers:** The spoken part of BNC is divided into speech turns, each accompanied

⁴<https://daringfireball.net/projects/markdown/>

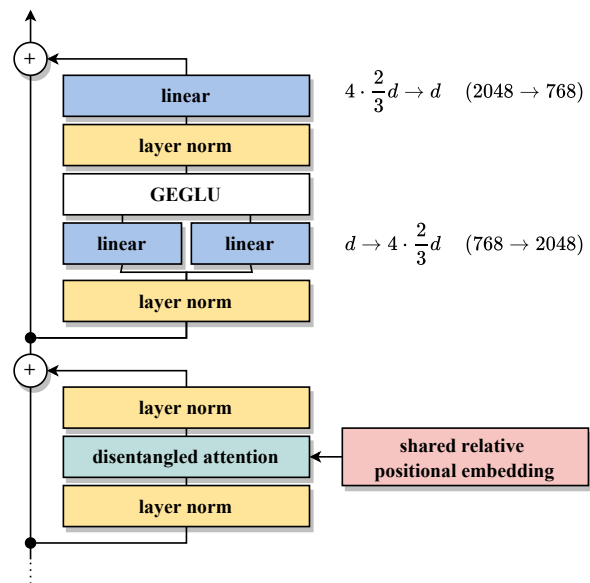


Figure 1: A simplified diagram of one layer in our LTG-BERT language model, which illustrates the changes made to the standard Transformer architecture – NormFormer layer normalization, GEGLU activation function and disentangled attention.

by a speaker identifier. We maintain this information by formatting each speech turn as ‘{name} : _␣ {turn}’.

3. **Quotes:** Markdown also allows us to keep the special quoted text by using a prefix ‘> _␣’.
4. **Lists:** The XML format contains special tags for lists and their respective elements, we use the ‘- _␣{element}’ notation to encode these text blocks.
5. **Incomprehensible speech:** Some words or phrases could not be transcribed because they were illegible or inaudible. Since completely omitting such text would result in ungrammatical sentences, we mark these segments with a special ‘[UNK]’ token.

Not all of this additional information is of use for the language models tested in this article, but it can be easily filtered out when needed. We preserve it to make this corpus more versatile.

4 Model architecture

We slightly depart from the typical *post-norm* Transformer architecture (Vaswani et al., 2017) used by BERT (Devlin et al., 2019), as illustrated in Figure 1. Preliminary experiments with this model showed that it tends to unpredictably diverge in

the later stages of training. This behavior has been noted in previous work on large LMs (Liu et al., 2020) and accordingly, we follow some of the recent improvements of Transformer.

NormFormer. *Pre-norm* variation of the Transformer has been shown to lead to more stable convergence with slightly degraded performance (Nguyen and Salazar, 2019). Shleifer and Ott (2022) claimed to mitigate this degradation by introducing an additional layer normalization operation. For these reasons, we decided to use their so-called *NormFormer* architecture to stabilize the training.⁵

GEGLU activation function, proposed in Shazeer (2020), enhances the expressiveness of the original Transformer feed-forward modules by redefining them as

$$\text{FF}_{\text{GEGLU}}(\mathbf{x}) = (\text{GELU}(\mathbf{x}\mathbf{W}_1) \odot \mathbf{x}\mathbf{W}_2) \mathbf{W}_3,$$

where \mathbf{W}_i are weight matrices⁶ and GELU is the Gaussian Error Linear Unit (Hendrycks and Gimpel, 2016). Note that this formulation involves three linear transformations instead of two, we therefore lower the intermediate hidden size by $2/3$ to keep the number of parameters the same.

Disentangled attention. The original Transformer formulation (Vaswani et al., 2017) fuses the content and positional information together in the first embedding layer and calculates the (un-normalized) attention score between each pair of tokens \mathbf{x}_i and \mathbf{x}_j as

$$A_{i,j} = \frac{\mathbf{Q}_i \mathbf{K}_j^\top}{\sqrt{d}},$$

where \mathbf{Q} and \mathbf{K} are the query-key linear transformations of \mathbf{x} .

He et al. (2021) proposed to *disentangle* the content and positional information. The content representations are incrementally built by the Transformer layers and the position is encoded by one shared relative positional embedding matrix $P \in \mathbb{R}^{(2L-1) \times d}$, where L is the maximal input length.⁷ This is supposed to offer greater expressivity as each layer can access these two parts directly. The attention scores are then calculated as a sum of

⁵They also proposed some additional improvements – *head scaling* and *residual scaling*, but we did not experience any performance benefits from these changes.

⁶The bias terms are omitted for brevity.

⁷Tokens at positions i and j have relative positional embedding at the $(L - i + j)$ th row of P , denoted as $P_{i,j}$.

three distinct parts: *content-to-content*, *content-to-position* and *position-to-content* attention – formally, the attention scores are defined as

$$A_{i,j} = \frac{{}^c\mathbf{Q}_i {}^c\mathbf{K}_j^\top + {}^c\mathbf{Q}_i {}^p\mathbf{K}_{i,j}^\top + {}^p\mathbf{Q}_{j,i} {}^c\mathbf{K}_j^\top}{\sqrt{3d}},$$

where ${}^c\mathbf{Q}$ and ${}^c\mathbf{K}$ are linear transformations of the *content* vectors and ${}^p\mathbf{Q}$ and ${}^p\mathbf{K}$ are linear transformations of the relative *positional* embedding $P_{i,j}$. We share the parameters of the content and positional transformations, ${}^c\mathbf{Q} = {}^p\mathbf{Q}$ and ${}^c\mathbf{K} = {}^p\mathbf{K}$, to not increase the model size while achieving comparable performance (He et al., 2021).

Initialization scaling. Bajaj et al. (2022) found that we can further stabilize the Transformer architecture by gradually scaling down its feed-forward (FF) weight matrices. Following Nguyen and Salazar (2019), we first initialize all weight matrices \mathbf{W} by sampling from:

$$\mathbf{W}_{i,j} \sim \mathcal{N}\left(0, \sqrt{\frac{2}{d + 4d}}\right),$$

where d is the hidden dimension.⁸ Then all three weight matrices in a FF module at layer l are scaled down by a factor of $1/\sqrt{2^{l+1}}$.

5 Training objectives

The fixed corpus, tokenizer and fine-tuning procedures establish a controlled test bed for a comparative study of training objectives proposed in the past. The original BERT model is trained via two self-supervised training objectives – masked language modeling (MLM) and next sentence prediction (NSP). We evaluate five different configurations of these objectives (three for MLM and two for NSP), as further detailed below.

5.1 Masked language modeling (MLM)

Unlike the traditional auto-regressive language models, the *Bidirectional* Encoder Representations from Transformers (BERT) learn a *bidirectional* contextualized representation for each token in a text segment. This is done by randomly selecting 15% of subword tokens (excluding the special tokens). Out of these, 80% are masked, 10% randomly replaced and 10% are left untouched. The

⁸This formula is roughly equal to the universal BERT initialization range of 0.02 for $d = 1024$.

language model is then trained to jointly predict the original state of the selected units. We investigate three common choices of the masked text units:

1. **Subwords.** As proposed in the seminal work by Devlin et al. (2019), every subword is masked independently with 15% probability to model its bidirectional dependencies.
2. **Whole words.** This method was also implemented by Devlin et al. (2019), after the publication of their original paper with subword masking. The motivation for this approach is that partially masked multi-subword word units are often easily decoded without any need for non-local contextual information; masking the whole multi-subword unit should force the model to build longer-range non-local dependencies.
3. **Spans.** The third method further follows the direction of whole-word masking and generalizes it to masking of random *spans* of subwords. More specifically, SpanBERT (Joshi et al., 2020) iteratively samples random spans until 15% of subwords are masked. For each span, it first samples its length from $\text{Geo}(p)$, where $p = 1/3$.⁹ Then the starting subword of the masked span is chosen from a uniform distribution.

5.2 Next sentence prediction (NSP)

Masked language modeling is a token-level training objective that trains the model to learn rich token representations. Yet, some downstream tasks need a single sentence-level representation instead. To also learn these, researchers have designed a number of additional semi-supervised training objectives. On the other hand, Liu et al. (2019) argue that NSP objectives do not help the downstream performance and they can thus be dropped in favour of a simpler optimization process with a single MLM training objective. To test these hypotheses, we experiment with two NSP objectives:

1. **Document discrimination.** Devlin et al. (2019) sample two text segments and then train the model with a second discriminative loss function, which predicts whether the two segments are continual or randomly taken from two different documents.

⁹ To ensure that the sampled length is not too large, we take the sampled value modulo 10. The expected length of a masked span is then approximately equal to 2 with $p = 1/3$.

2. **Sentence-order discrimination.** Lan et al. (2020) argue that the document discrimination is too easy as the language models only have to compare the topic of the two segments to achieve a good performance in this task. Instead, they propose to predict whether the two segments are in the correct order or whether they are swapped. Thus, the sentence-order loss forces the neural network to model inter-sentence coherence and this is believed to lead to a better downstream performance.

6 Evaluation metrics

We use three conceptually different methods for evaluating the amount of linguistic knowledge acquired by the BNC language models. 1) The (Super)GLUE datasets test the ability of the model to adapt to various NLU tasks by further optimizing the whole pre-trained model, 2) edge probing tasks evaluate how much linguistic information one can extract from a frozen pre-trained model and 3) BLiMP utilizes the intrinsic ability of the pre-trained network to model language and probes its knowledge without any additional training. We further elaborate on each of these below.

6.1 (Super)GLUE

GLUE (Wang et al., 2018) and SuperGLUE (Wang et al., 2019) have become a de-facto standard for evaluating the language understanding capabilities of language models. Accordingly, we also choose to fine-tune our language models on these NLU tasks to measure their linguistic and transfer-learning performance. We give more technical details about our implementation of (Super)GLUE fine-tuning in Appendix B.1.

We exclude the Winograd schema datasets, WNLI and WSC, because they require a complete reformulation to get past the trivial most-frequent baseline (Kocijan et al., 2019). The remaining 14 (Super)GLUE datasets measure performance on these tasks:

- **Inference:** CB, MNLI, QNLI, RTE.
- **Linguistic acceptability:** CoLA.
- **Sentiment analysis:** SST-2.
- **Semantic similarity:** MRPC, QQP, STS-B.
- **Word sense disambiguation:** WiC.
- **Question answering:** BoolQ, COPA, MultiRC, ReCoRD.

6.1.1 HANS

Deep learning systems are (by design) prone to finding spurious correlations in the training data. These heuristics can often be successfully employed for the evaluation data, as well – thus, one has to be careful when implying that a higher score on a benchmark shows a deeper understanding of the tested model. McCoy et al. (2019) tried to evaluate to what extent language models rely on spurious heuristics to solve NLI tasks. They identified a set of fallible syntactic heuristics and designed a test set where these ‘shortcuts’ should fail – Heuristic Analysis for NLI Systems (HANS). We adopt their approach and test models that have been fine-tuned on MNLI.

6.2 Edge probing

GLUE tasks measure the ability of a LM to be fine-tuned on a sentence-level NLU problem. To get a more comprehensive picture of LM performance, one can also *probe* the word-level contextualized representations, measuring how much syntactic or semantic information can be extracted.

Tenney et al. (2019) devised a simple approach of probing for a diverse set of linguistic phenomena called *edge probing*. They reformulate traditional NLP tasks as *span classification*: part-of-speech tagging can be viewed as classification of word-spans and semantic role labeling becomes a classification of pairs of spans: predicate-span and argument-span. In the following, we will probe our models for five basic tasks: part-of-speech tagging (POS), dependency parsing (DP), semantic role labeling (SRL), named-entity recognition (NER) and coreference resolution (CR). Note that the model only learns to *classify* each span provided to the model as gold data. This substantially simplifies some of the tasks, for example SRL. Please refer to Appendix B.2 for the implementation details of edge probing.

6.3 BLiMP

One disadvantage of the aforementioned evaluation metrics is that the results are skewed by the second-stage supervised training, which makes it problematic to disentangle the prior knowledge of a language model from the acquired knowledge (Belinkov, 2022). In contrast, the Benchmark of Linguistic Minimal Pairs (BLiMP; Warstadt et al., 2020) attempts to measure the linguistic knowledge of a language model in a zero-shot manner – with

out any additional training. The dataset consists of 67 000 sentence pairs; each pair differs minimally on the surface level, but only one of the sentences is grammatically valid. We can use the intrinsic ability of language models to assign a probability to every sentence and test how often a language model assigns a higher probability to the correct sentence. Appendix B.3 gives more details about ranking the likelihood of sentences according to the raw output of a masked language model.

7 Experiments

We conduct a number of experiments in this section. First, we compare different training hyperparameters and model configurations described in Section 4. Then, using the overall best training setting, we make a comparative study of training objectives (Section 5). Finally, we investigate the sampling efficiency of our proposed language model and we compare BNC with a Wikipedia & BookCorpus subset of the same size. These results can then be used as a baseline performance of BNC-BERT in future studies.

The central model used in the experiments is a *base-sized* Transformer – 12 encoder layers with hidden size 768 and 12 attention heads (more details in Appendix E). All reported models utilize the same cased WordPiece tokenizer (Wu et al., 2016) with a vocabulary size of $2^{14} = 16\,384$ trained with the BNC dataset (Appendix G). This goes against the trend of increasing the subword vocabulary in recent work,¹⁰ but a larger vocabulary size would lead to a lot of infrequent tokens within our limited corpus – we roughly follow Gowda and May (2020) and ‘... use the largest possible BPE vocabulary such that at least 95% of classes have 100 or more examples in training.’

Since our aim is to train models comparable to BERT_{base}, we train for the same amount of sampled tokens. Devlin et al. (2019) trained on 1M batches of 128K tokens, we use 31 250 training steps with batch size of 4M tokens to parallelize and accelerate the process. Also, similarly to Devlin et al. (2019), we use sequence length of 128 tokens in the first 90% of training and a larger sequence length of 512 only in the last 10% of steps. We deliberately do not compare against more recent models, which are trained for much longer to achieve slightly bet-

¹⁰ BERT (Devlin et al., 2019) uses 28 996 tokens, RoBERTa (Liu et al., 2019) 50 265 and in 2021, DeBERTa (He et al., 2021) used a vocabulary of 128 100 subwords.

ter performance: RoBERTa is trained on $16\times$ more training samples, for example.¹¹

7.1 Comparison of model architectures and training settings

In order to establish a strong baseline, we evaluate the proposed changes from Section 4 and other training configurations. We present the results in Table 2, where we compare the final model with all changes applied and models with one of those modifications removed. These training choices turned out to be the most important:

- Both the post-norm and pre-norm transformer variants perform substantially worse than the NormFormer-like layer normalization (Shleifer and Ott, 2022). Both of them also lead to less stable and only slightly faster training.
- Absolute positional embeddings seem to be less adaptable for fine-tuning but perform better on language modeling itself, as can be seen on the BLiMP results. We hypothesize that this is caused by more accurate estimation of probabilities of the first few words in a sentence. The simpler absolute embeddings also lead to the greatest reduction of training time. We choose the slower relative positional embeddings despite this fact to increase the performance on (Super)GLUE tasks.
- We observe that setting the weight decay correctly is crucial for masked language modeling. The default weight decay value found in Devlin et al. (2019), 0.01, performs substantially worse on all tested tasks. We use a higher decay value of 0.1 to boost performance, this value is most likely strongly correlated with the corpus size we use here. This suggests that previous findings of inferior performance of LMs pre-trained on small corpora might be caused by insufficient hyperparameter search.
- As expected, the AdamW optimizer (Loshchilov and Hutter, 2019) behaves poorly in our highly parallel training regime. Our study successfully replicates the reported performance of the LAMB optimizer (You et al., 2020), which we thus use in all other experiments.

¹¹ 500K steps with 8 192 segments of length 512, according to (He et al., 2021).

| Model | MNLI | Edge probing | BLiMP | Training time |
|-----------------------|-----------------------|-----------------------|-------------|------------------|
| LTG-BERT | 85.1 \pm 0.2 | 95.3 \pm 0.1 | 83.4 | 8h 13min |
| w/ post-norm (0.005) | -0.5 \pm 0.2 | -0.6 \pm 0.1 | -0.1 | -22min |
| w/ pre-norm (0.005) | -1.3 \pm 0.1 | -0.2 \pm 0.1 | -0.9 | -35min |
| w/ GELU activation | -0.3 \pm 0.3 | 0.0 \pm 0.1 | -0.1 | -6min |
| w/ absolute pos. emb. | -1.1 \pm 0.2 | -0.1 \pm 0.1 | +0.6 | -2h 16min |
| w/o FF init. scaling | -0.3 \pm 0.2 | -0.1 \pm 0.1 | +0.1 | 0min |
| w/ learnt FF biases | -0.3 \pm 0.2 | 0.0 \pm 0.1 | -0.1 | +9min |
| w/ 0.01 WD (0.005) | -1.4 \pm 0.1 | -0.2 \pm 0.1 | -0.7 | -1min |
| w/ linear schedule | -0.5 \pm 0.2 | 0.0 \pm 0.1 | -0.2 | 0min |
| w/ AdamW (0.001) | -0.9 \pm 0.2 | -0.2 \pm 0.1 | -0.5 | -11min |

Table 2: Comparative study of different architectural and training settings. The first row shows the performance of the final model with all improvements applied and the following rows give the relative changes in performance when one of the changes is not applied – for example, the second row tests swapping the NormFormer-like normalization with the ‘post-norm’ normalization. Some runs diverged with the default learning rate of 0.01 and had to be run again with a lower value (denoted in parentheses). ‘WD’ stands weight decay and ‘FF’ is an abbreviation for the feed-forward modules. We report the mean and standard deviation statistics across five runs, if applicable, and boldface all run within 1 standard deviation from the best result.

The other changes bring more marginal gains – all three tested modifications of the feed-forward layers work slightly better: 1) using GEGLU activation function instead of GELU, 2) initializing the feed-forward layers with incrementally lower weight norms, and 3) not using any bias parameters in these layers. The last tested change shows that cosine learning rate decay (Rae et al., 2021) performs better than the standard linear weight decay.

7.2 Training objective comparison

Masked language modeling. First of all, we compare the three masking methods described in Section 5.1: subword, whole-word and span masking. The summary of the results is given in Table 3, more detailed evaluation in Appendix D. Overall, the span-based masking performs marginally better than the other methods – it shows a clear improvement on (Super)GLUE benchmarks over the simple subword masking, it generalizes the best according to the HANS score and it even matches the performance of BERT_{base} on the averaged BLiMP accuracy. All methods perform equally well on edge probing. Whole-word masking lacks on the BLiMP benchmark because the model is not expecting partially masked words that can occur in

| Model (variant) | GLUE | | | | | HANS | Edge probing | BLiMP |
|---|--------------------------------|--------------------------------|--------------------------------|--------------------------------|--------------------------------|----------------|--------------------------------|-------------|
| | MNLI | MRPC | QNLI | SST-2 | Average | | | |
| Wikipedia + BookCorpus (3000M words; Devlin et al., 2019) | | | | | | | | |
| BERT _{base, cased} [†] | 84.4 | 86.7 | 88.4 | 92.7 | 88.1 | 69.0 | 93.9 | 84.2 |
| BERT _{base, cased} (our eval.) | 83.6 \pm 0.2 | 84.6 \pm 0.5 | 90.8 \pm 0.1 | 91.9 \pm 0.4 | 87.8 \pm 0.3 | 61.8 \pm 1.5 | 93.8 \pm 0.2 | 84.2 |
| Wikipedia + BookCorpus (100M words) | | | | | | | | |
| LTG-BERT (subword masking) | 84.2 \pm 0.1 | 84.3 \pm 0.7 | 90.8 \pm 0.3 | 92.1 \pm 0.5 | 87.8 \pm 0.5 | 62.5 \pm 1.7 | 95.3\pm0.1 | 82.0 |
| British National Corpus (100M words) | | | | | | | | |
| LTG-BERT (subword masking) | 85.1\pm0.2 | 85.0 \pm 0.9 | 90.0 \pm 0.3 | 92.7\pm0.4 | 88.2 \pm 0.5 | 64.4 \pm 1.3 | 95.3\pm0.1 | 83.4 |
| LTG-BERT (whole-word masking) | 84.9 \pm 0.2 | 85.5 \pm 0.9 | 90.6 \pm 0.3 | 92.7\pm0.2 | 88.4 \pm 0.5 | 63.7 \pm 0.8 | 95.3\pm0.1 | 80.1 |
| LTG-BERT (span masking) | 85.1\pm0.2 | 87.5\pm0.9 | 91.5\pm0.2 | 92.8\pm0.5 | 89.2\pm0.5 | 65.6 \pm 0.5 | 95.2\pm0.1 | 84.2 |
| LTG-BERT (subword + document NSP) | 85.2\pm0.3 | 86.5 \pm 0.8 | 90.3 \pm 0.2 | 92.2 \pm 0.4 | 88.6 \pm 0.5 | 60.5 \pm 1.2 | 95.3\pm0.1 | 83.3 |
| LTG-BERT (subword + order NSP) | 84.7 \pm 0.1 | 85.9 \pm 0.6 | 90.4 \pm 0.2 | 92.1 \pm 0.2 | 88.3 \pm 0.4 | 64.2 \pm 1.9 | 95.1 \pm 0.1 | 82.2 |
| LTG-BERT (subword + 2 \times steps) | 85.2\pm0.2 | 86.5 \pm 0.8 | 90.3 \pm 0.3 | 92.3\pm0.6 | 88.6 \pm 0.5 | 65.3 \pm 1.1 | 95.3\pm0.1 | 83.5 |
| LTG-BERT (subword + 1/2 \times steps) | 84.4 \pm 0.3 | 86.3 \pm 1.1 | 90.4 \pm 0.2 | 92.8\pm0.4 | 88.5 \pm 0.6 | 62.4 \pm 0.8 | 95.2\pm0.1 | 83.5 |
| LTG-BERT (subword + 1/4 \times steps) | 83.8 \pm 0.2 | 85.3 \pm 0.8 | 89.1 \pm 0.2 | 91.7 \pm 0.4 | 87.5 \pm 0.5 | 58.6 \pm 1.3 | 95.0 \pm 0.1 | 83.2 |
| Random initialization | 59.5 \pm 0.5 | 68.5 \pm 1.4 | 63.8 \pm 0.2 | 82.2 \pm 0.7 | 68.5 \pm 0.8 | 49.7 \pm 0.3 | 73.1 \pm 0.4 | 50.0 |

Table 3: Summary of the experimental results. We show the results on the 4 GLUE tasks with known development results from Devlin et al. (2019) and their average; then the accuracy on HANS, the average of all 5 edge probing tasks and 67 BLiMP tasks. [†]The BERT_{base, cased} results are shown primarily for reference, they come from these sources: partial development GLUE scores from Devlin et al. (2019), edge probing from Tenney et al. (2019), HANS from Bhargava et al. (2021) and BLiMP from Salazar et al. (2020). We also add the BERT_{base, cased} results from our evaluation scripts for more fair and accurate comparison. We present the mean and standard deviation statistics over 5 evaluation runs and boldface all run within 1 standard deviation from the best result. The detailed results can be found in Appendix D.

the evaluation (Section 6.3). The original subword masking strategy is still a competitive baseline and it might be preferred in practice due to its simple implementation.

Next-sentence prediction. Next, we experiment with combining an NSP task and simple subword masking. We hypothesize that a second training objective might extract more information from the limited BNC corpus, which would help with the downstream performance – an opposite conclusion than Liu et al. (2019). However, our hypothesis turns out to be wrong, according to the results in Table 3. The experiments agree with the design of latest masked language models – next sentence prediction is an unnecessary training objective, at least for the tasks evaluated in this paper. It does not lead to substantially improved sentence representations even in a limited data regime. We can also see that the well-motivated order discrimination (Lan et al., 2020), proposed to solve the issues of document discrimination, actually leads to an overall worse performance. Hence we cannot recommend to complicate pre-training with a second training objective.

7.3 Sampling efficiency

An important aspect of efficient language models is the number of training steps they require to reach a sufficient performance. So far, we have limited the size of the training corpus but kept the number of steps constant, set according to Devlin et al. (2019). The results in Table 3 suggest that increasing the steps two times does not lead to a noticeably better performance with BNC. Even more so, training for half the time turns out to be enough to get comparable performance. Yet, decreasing the training steps further starts to degrade the downstream results too much, as evidenced by the scores obtained with 1/4 of the default steps.

These results highlight the sampling inefficiency of current self-supervised language modeling methods, as even with 1/4 steps, every token in BNC is seen about roughly 250 times during training.¹² We hope that a future work in this field will be able to learn from a smaller number of samples.

¹² This value can be calculated from Table 9: these models are trained for 7 812 steps with 4 194 304 tokens per batch. Table 1 shows that there are 131 392 103 subwords in the BNC train split.

7.4 100 million subset of Wikipedia & BookCorpus

Our last experiment evaluates how much does the careful curation of BNC help the downstream performance. To keep the comparability to BERT, we choose to pre-train on a random subset of Wikipedia and BookCorpus (with equal size to BNC, sampled document-wise); this corpus is constructed according to [Appendix F](#). Note that BNC is a corpus of British English compiled in 1990s so some evaluation tasks can be skewed against it – for example QNLI, which is based on texts from Wikipedia. [Table 3](#) shows that a high-quality data source is not necessarily needed to learn from 100M words but better quality leads to a noticeable difference in downstream performance.

8 Conclusion

In this paper, we evaluated how data-efficient masked language models can be. In particular, we trained a variety of models with different training objectives on the same training data: British National Corpus. Although small by modern standards (100M tokens), it is well balanced and carefully crafted to represent British English of the 20th century. On a variety of benchmarks, our models perform better than BERT_{base} trained on a much larger corpus. We believe that this limited data regime is beneficial for the development of efficient and reliable language models. Our finding also suggests that 100 million word tokens is enough to learn basic linguistic skills by current language modeling techniques, given that the data is carefully selected and balanced. To conclude, huge amounts of training data are not always necessary – we should focus on more efficient training settings instead.

We showed that the next sentence prediction objective does not improve BERT-like models, confirming the findings in [Liu et al. \(2019\)](#). In addition, the standard subword masking from [Devlin et al. \(2019\)](#) is consistently outperformed by the span masking method and the linguistic performance can be substantially increased by utilizing better neural architectures and training configurations. We release the code for training and using BERT-like models with the optimal architectural choices (according to our experiments) under the name LTG-BERT.¹³

¹³<https://github.com/ltgoslo/ltg-bert>

The presented results serve primarily as the foundation for future research on efficient language modeling. We hope our work shows the value of careful curation of representative corpora and will spark more interest in this area, where BNC can serve as an undemanding, replicable and openly-available training corpus.

9 Limitations

First of all, our work only considers language modeling of English and does not provide results on any other language – even though we hope that our conclusions could be useful for low-resource languages. Secondly, even though we found out that it is possible to train a competent language model with a small corpus, the training process still requires a similar amount of computational resources to models trained with larger corpora, as noted in [Section 7.3](#). Finally, we evaluate mainly the linguistic knowledge of language models ([Section 6](#)), our conclusions might not apply for their general knowledge.

References

- Yeşim Aksan, Mustafa Aksan, Ahmet Koltuksuz, Taner Sezer, Ümit Mersinli, Umut Ufuk Demirhan, Hakan Yılmaz, Gülsüm Atasoy, Seda Öz, İpek Yıldız, and Özlem Kurtoğlu. 2012. [Construction of the Turkish national corpus \(TNC\)](#). In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pages 3223–3227, Istanbul, Turkey. European Language Resources Association (ELRA). 3
- Giuseppe Attardi. 2015. Wikiextractor. <https://github.com/attardi/wikiextractor>. 17
- Payal Bajaj, Chenyan Xiong, Guolin Ke, Xiaodong Liu, Di He, Saurabh Tiwary, Tie-Yan Liu, Paul Bennett, Xia Song, and Jianfeng Gao. 2022. [Metro: Efficient denoising pretraining of large scale autoencoding language models with model generated signals](#). 4
- Roy Bar-Haim, Ido Dagan, Bill Dolan, Lisa Ferro, and Danilo Giampiccolo. 2006. The second pascal recognising textual entailment challenge. *Proceedings of the Second PASCAL Challenges Workshop on Recognising Textual Entailment*. 16
- Yonatan Belinkov. 2022. [Probing Classifiers: Promises, Shortcomings, and Advances](#). *Computational Linguistics*, 48(1):207–219. 6
- Emily M Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. 2021. On the dangers of stochastic parrots: Can language models

- be too big? In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, pages 610–623. 1
- Luisa Bentivogli, Ido Dagan, Hoa Trang Dang, Danilo Giampiccolo, and Bernardo Magnini. 2009. The fifth pascal recognizing textual entailment challenge. In *In Proc Text Analysis Conference (TAC'09)*. 16
- Prajjwal Bhargava, Aleksandr Drozd, and Anna Rogers. 2021. [Generalization in NLI: Ways \(not\) to go beyond simple heuristics](#). In *Proceedings of the Second Workshop on Insights from Negative Results in NLP*, pages 125–135, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics. 8
- Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. [Language models are few-shot learners](#). In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc. 2
- Lou Burnard. 2002. *Where did we Go Wrong? A Retrospective Look at the British National Corpus*, pages 51 – 70. Brill, Leiden, The Netherlands. 2
- Daniel Cer, Mona Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. 2017. [SemEval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation](#). In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 1–14, Vancouver, Canada. Association for Computational Linguistics. 16
- Ciprian Chelba, Tomas Mikolov, Mike Schuster, Qi Ge, Thorsten Brants, Phillipp Koehn, and Tony Robinson. 2014. [One billion word benchmark for measuring progress in statistical language modeling](#). In *Proc. Interspeech 2014*, pages 2635–2639. 2
- Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. [BoolQ: Exploring the surprising difficulty of natural yes/no questions](#). In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2924–2936, Minneapolis, Minnesota. Association for Computational Linguistics. 15
- BNC Consortium. 2007. British National Corpus. 1, 3
- Ido Dagan, Oren Glickman, and Bernardo Magnini. 2006. The pascal recognising textual entailment challenge. In *Machine Learning Challenges. Evaluating Predictive Uncertainty, Visual Object Classification, and Recognising Tectual Entailment*, pages 177–190, Berlin, Heidelberg. Springer Berlin Heidelberg. 16
- Marie-Catherine de Marneffe, Mandy Simons, and Judith Tonhauser. 2019. [The commitmentbank: Investigating projection in naturally occurring discourse](#). *Proceedings of Sinn und Bedeutung*, 23(2):107–124. 15
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. [BERT: Pre-training of deep bidirectional transformers for language understanding](#). In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics. 2, 3, 5, 5, 5, 6, 6, 6, 7, 8, 8, 8, 9
- William B. Dolan and Chris Brockett. 2005. [Automatically constructing a corpus of sentential paraphrases](#). In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*. 15
- Danilo Giampiccolo, Bernardo Magnini, Ido Dagan, and Bill Dolan. 2007. [The third PASCAL recognizing textual entailment challenge](#). In *Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing*, pages 1–9, Prague. Association for Computational Linguistics. 16
- Thamme Gowda and Jonathan May. 2020. [Finding the optimal vocabulary size for neural machine translation](#). In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3955–3964, Online. Association for Computational Linguistics. 6, 17
- Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2021. [DeBERTa: Decoding-enhanced bert with disentangled attention](#). In *International Conference on Learning Representations*. 4, 4, 6, 7
- Dan Hendrycks and Kevin Gimpel. 2016. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*. 4
- Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katie Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, Jack W. Rae, Oriol Vinyals, and Laurent Sifre. 2022. [Training compute-optimal large language models](#). 2
- F. Jelinek. 1976. [Continuous speech recognition by statistical methods](#). *Proceedings of the IEEE*, 64(4):532–556. 2

- Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Tuo Zhao. 2020. [SMART: Robust and efficient fine-tuning for pre-trained natural language models through principled regularized optimization](#). In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2177–2190, Online. Association for Computational Linguistics. 14
- Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S. Weld, Luke Zettlemoyer, and Omer Levy. 2020. [SpanBERT: Improving pre-training by representing and predicting spans](#). *Transactions of the Association for Computational Linguistics*, 8:64–77. 5
- 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. 16
- Vid Kocijan, Ana-Maria Cretu, Oana-Maria Camburu, Yordan Yordanov, and Thomas Lukasiewicz. 2019. [A surprisingly robust trick for the Winograd schema challenge](#). In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4837–4842, Florence, Italy. Association for Computational Linguistics. 5
- Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. [Albert: A lite bert for self-supervised learning of language representations](#). In *International Conference on Learning Representations*. 5, 8
- Tal Linzen. 2020. [How can we accelerate progress towards human-like linguistic generalization?](#) In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5210–5217, Online. Association for Computational Linguistics. 1
- Liyuan Liu, Xiaodong Liu, Jianfeng Gao, Weizhu Chen, and Jiawei Han. 2020. [Understanding the difficulty of training transformers](#). In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5747–5763, Online. Association for Computational Linguistics. 4
- Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. [Roberta: A robustly optimized BERT pretraining approach](#). *CoRR*, abs/1907.11692. 2, 5, 6, 8, 9
- Ilya Loshchilov and Frank Hutter. 2019. [Decoupled weight decay regularization](#). In *International Conference on Learning Representations*. 7
- B.W. Matthews. 1975. [Comparison of the predicted and observed secondary structure of t4 phage lysozyme](#). *Biochimica et Biophysica Acta (BBA) - Protein Structure*, 405(2):442–451. 15
- Tom McCoy, Ellie Pavlick, and Tal Linzen. 2019. [Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference](#). In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3428–3448, Florence, Italy. Association for Computational Linguistics. 6
- Toan Q. Nguyen and Julian Salazar. 2019. [Transformers without tears: Improving the normalization of self-attention](#). In *Proceedings of the 16th International Conference on Spoken Language Translation*, Hong Kong. Association for Computational Linguistics. 4, 4
- Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. [Pytorch: An imperative style, high-performance deep learning library](#). In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems* 32, pages 8024–8035. Curran Associates, Inc. 17
- Matthew E. 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. 2
- Mohammad Taher Pilehvar and Jose Camacho-Collados. 2019. [WiC: the word-in-context dataset for evaluating context-sensitive meaning representations](#). In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1267–1273, Minneapolis, Minnesota. Association for Computational Linguistics. 16
- Jack W. Rae, Sebastian Borgeaud, Trevor Cai, Katie Millican, Jordan Hoffmann, Francis Song, John Aslanides, Sarah Henderson, Roman Ring, Susannah Young, Eliza Rutherford, Tom Hennigan, Jacob Menick, Albin Cassirer, Richard Powell, George van den Driessche, Lisa Anne Hendricks, Maribeth Rauh, Po-Sen Huang, Amelia Glaese, Johannes Welbl, Sumanth Dathathri, Saffron Huang, Jonathan Uesato, John Mellor, Irina Higgins, Antonia Creswell, Nat McAleese, Amy Wu, Erich Elsen, Siddhant Jayakumar, Elena Buchatskaya, David Budden, Esme Sutherland, Karen Simonyan, Michela Paganini, Laurent Sifre, Lena Martens, Xiang Lorraine Li, Adhiguna Kuncoro, Aida Nematzadeh, Elena Gribovskaya, Domenic Donato, Angeliki Lazaridou,

- Arthur Mensch, Jean-Baptiste Lespiau, Maria Tsim-poukelli, Nikolai Grigorev, Doug Fritz, Thibault Sot-tiaux, Mantas Pajarskas, Toby Pohlen, Zhitao Gong, Daniel Toyama, Cyprien de Masson d'Autume, Yujia Li, Tayfun Terzi, Vladimir Mikulik, Igor Babuschkin, Aidan Clark, Diego de Las Casas, Aurelia Guy, Chris Jones, James Bradbury, Matthew Johnson, Blake Hechtman, Laura Weidinger, Iason Gabriel, William Isaac, Ed Lockhart, Simon Osindero, Laura Rimell, Chris Dyer, Oriol Vinyals, Kareem Ayoub, Jeff Stanway, Lorraine Bennett, Demis Hassabis, Ko-ray Kavukcuoglu, and Geoffrey Irving. 2021. [Scaling language models: Methods, analysis & insights from training gopher](#). 2, 7
- 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*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics. 16
- Melissa Roemmele, Cosmin Bejan, and Andrew Gordon. 2011. Choice of plausible alternatives: An evaluation of commonsense causal reasoning. *AAAI Spring Symposium - Technical Report*. 15
- Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2020. [A primer in BERTology: What we know about how BERT works](#). *Transactions of the Association for Computational Linguistics*, 8:842–866. 15
- Julian Salazar, Davis Liang, Toan Q. Nguyen, and Ka-trin Kirchhoff. 2020. [Masked language model scoring](#). In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2699–2712, Online. Association for Computational Linguistics. 8, 14
- Noam Shazeer. 2020. [GLU variants improve trans-former](#). *CoRR*, abs/2002.05202. 4
- Sam Shleifer and Myle Ott. 2022. [Normformer: Im-proved transformer pretraining with extra normaliza-tion](#). 4, 7
- Natalia Silveira, Timothy Dozat, Marie-Catherine de Marneffe, Samuel Bowman, Miriam Connor, John Bauer, and Christopher D. Manning. 2014. A gold standard dependency corpus for English. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC-2014)*. 16
- 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 *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics. 16
- Ian Tenney, Patrick Xia, Berlin Chen, Alex Wang, Adam Poliak, R Thomas McCoy, Najoung Kim, Benjamin Van Durme, Sam Bowman, Dipanjan Das, and Ellie Pavlick. 2019. [What do you learn from con-text? probing for sentence structure in contextualized word representations](#). In *International Conference on Learning Representations*. 6, 8, 14, 16, 18
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. [Attention is all you need](#). In *Advances in Neural Information Pro-cessing Systems*, volume 30. Curran Associates, Inc. 2, 3, 4, 14
- Alex Wang and Kyunghyun Cho. 2019. [BERT has a mouth, and it must speak: BERT as a Markov ran-dom field language model](#). In *Proceedings of the Workshop on Methods for Optimizing and Evaluat-ing Neural Language Generation*, pages 30–36, Min-neapolis, Minnesota. Association for Computational Linguistics. 14
- Alex Wang, Yada Pruksachatkun, Nikita Nangia, Aman-preet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2019. [Superglue: A stickier benchmark for general-purpose language understand-ing systems](#). In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc. 5
- Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. [GLUE: A multi-task benchmark and analysis platform for nat-ural language understanding](#). In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Com-putational Linguistics. 5
- Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mo-hananey, Wei Peng, Sheng-Fu Wang, and Samuel R. Bowman. 2020. [BLiMP: The benchmark of linguis-tic minimal pairs for English](#). *Transactions of the As-sociation for Computational Linguistics*, 8:377–392. 6, 16, 16
- Alex Warstadt, Amanpreet Singh, and Samuel R. Bow-man. 2019. [Neural network acceptability judgments](#). *Transactions of the Association for Computational Linguistics*, 7:625–641. 15
- Weischedel, Ralph, Palmer, Martha, Marcus, Mitchell, Hovy, Eduard, Pradhan, Sameer, Ramshaw, Lance, Xue, Nianwen, Taylor, Ann, Kaufman, Jeff, Franchini, Michelle, El-Bachouti, Mohammed, Belvin, Robert, and Houston, Ann. 2013. [Ontonotes release 5.0](#). 16
- Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. [A broad-coverage challenge corpus for sen-tence understanding through inference](#). In *Proceed-ings of the 2018 Conference of the North American Chapter of the Association for Computational Lin-guistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122, New Orleans, Louisiana. Association for Computational Linguistics. 15

- Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. [Transformers: State-of-the-art natural language processing](#). In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics. 17
- Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Łukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. [Google’s neural machine translation system: Bridging the gap between human and machine translation](#). 6
- Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. [Xlnet: Generalized autoregressive pretraining for language understanding](#). In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc. 2
- Yang You, Jing Li, Sashank Reddi, Jonathan Hseu, Sanjiv Kumar, Srinadh Bhojanapalli, Xiaodan Song, James Demmel, Kurt Keutzer, and Cho-Jui Hsieh. 2020. [Large batch optimization for deep learning: Training bert in 76 minutes](#). In *International Conference on Learning Representations*. 7
- 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](#). 16
- Yian Zhang, Alex Warstadt, Xiaocheng Li, and Samuel R. Bowman. 2021. [When do you need billions of words of pretraining data?](#) In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1112–1125, Online. Association for Computational Linguistics. 2
- Yukun Zhu, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. [Aligning books and movies: Towards story-like visual explanations by watching movies and reading books](#). In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 19–27. 2, 2, 17

A BNC samples

We follow the description of the Markdown conversion of BNC from Section 3.1 and show samples of the resulting raw Markdown text to illustrate this process, highlighting some of the formatting information captured by our format. A sample of a spoken document is given in Listing 1 and a sample of a written article is shown below, in Listing 2.

B Evaluation metrics – implementation details

B.1 (Super)GLUE

Fine-tuning of the GLUE and SuperGLUE tasks follows a straightforward framework: the segments are tokenized, concatenated – starting with a special [CLS] token and with a [SEP] token put in between the segments – and input to a pre-trained language model. Subsequently, the contextualized representation of the special [CLS] token is fed into an MLP classifier. The pre-trained weights are further fine-tuned together with the classifier weights.

We do not employ any additional training tricks that were used in the previous works to seemingly increase the performance of their large language models – e.g. further ‘pre-training’ on MNLI, multi-task learning, ensembling, extensive hyperparameter search, selecting the best random seeds, reformulating the tasks or complex regularization techniques such as SMART (Jiang et al., 2020).

B.2 Edge probing

We follow the description of edge probing in the original paper by Tenney et al. (2019). First of all, subword representations $s_{i,k}$ are extracted from a frozen LM, for all positions i and layers k . These are downsampled to a dimensionality of 256 by a linear transformation. To get a vector representation h_t for the t^{th} span, we apply two pooling operations on the subword-token representations $s_{t,k}$. First, we pool the vectors at all layers k by taking a learnt convex combination $\hat{s}_t = \sum_{k=1}^{12} \gamma_k s_{t,k}$, where $\gamma_k \in \mathbb{R}$. Next, since one span can be split into multiple subwords, we employ an attention pooling operator to get the span-level embeddings: $h_t = \sum_{i \in \mathcal{I}_t} \text{att}(\hat{s}_t; \theta) \hat{s}_t$, where \mathcal{I}_t are the subwords indices of the t^{th} span. Finally, the pooled vectors h_t are fed into a multi-layer perceptron (MLP) and classified. If a task requires a pair of span representations (DP, SRL and CR), then these are pooled

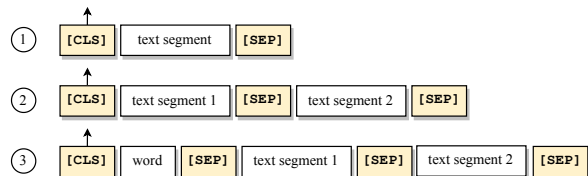


Figure 2: Three variations of (Super)GLUE input: 1) single-sentence tasks SST-2 and CoLA; 2) classification of a pair of text segments: BoolQ, CB, COPA, MNLI, MRPC, QNLI, QQP, STS-B, RTE; and 3) WiC (in the figure), MultiRC, ReCoRD.

with two separate attention operators and concatenated before being passed to the MLP classifier.

B.3 BLiMP

These models are trained to estimate $P(s_t | s_{<t})$ for sentence s and token s_t where $s_{<t} = (s_i | i < t)$; then the sentence log-probability is given by $\log P(s) = \sum_{t=1}^N \log P(s_t | s_{<t})$.

The issue with masked language models is that they are not designed to calculate this property; they are trained to estimate $P(s_t | s_{\setminus t})$ – the likelihood of a token s_t given its bidirectional context $s_{\setminus t} = (s_i | i \neq t)$. We can however still use MLMs to infer a *score* for each sentence where a higher *score* corresponds to a more likely sentence. Wang and Cho (2019) defined *pseudo-log-likelihood score* of a sentence s with model θ as

$$\text{PLL}(s) = \sum_{t=1}^N \log P(s_t | s_{\setminus t}; \theta).$$

Salazar et al. (2020) tested PLL and found that it produces accurate predictions on BLiMP. We adopt their approach and evaluate our models with PLL.

C Layer interpretation

The definition of the fine-tuning scheme for edge probing makes it straightforward to rate the contribution of each Transformer layer to a particular task – we can simply have a look at the layer-wise weights γ_k , see Table 5. To be more precise, if we define the k^{th} attention layer as a_k , the k^{th} feed-forward layer as ff_k and layer normalization operator as LN, then the k^{th} layer ℓ_k of a post-norm Transformer (Vaswani et al., 2017) computes the following function:

$$\begin{aligned} \hat{a}_k(x) &= \text{LN}(x + a_k(x)) \\ \ell_k &= \text{LN}(\hat{a}_k(\ell_{k-1}) + ff_k(\hat{a}_k(\ell_{k-1}))) \end{aligned}$$

| Task | BoolQ | CB | CoLA | COPA | MNLI | MRPC | MultiRC | QNLI | QQP | ReCoRD | RTE | SST2 | STSB | WiC |
|---------------------|-------|-----|-------|------|---------|-------|---------|---------|---------|-----------|-------|-------|-------|------|
| Train size | 9 427 | 250 | 8 551 | 800 | 392 702 | 3 668 | 27 243 | 104 743 | 363 846 | 1 179 400 | 2 490 | 67349 | 5749 | 5428 |
| Validation size | 3 270 | 56 | 1 043 | 100 | 9 815 | 408 | 4 848 | 5 463 | 40 430 | 113 236 | 277 | 872 | 1 500 | 638 |
| ≥ 512 subwords | 0.37% | 0% | 0% | 0% | 0% | 0% | 27.68% | 0.02% | 0% | 0.30% | 0% | 0% | 0% | 0% |

Table 4: The train and validation sizes of GLUE and SuperGLUE tasks (omitting WNLI and WSC). Note that we list the numbers of examples in the (Super)GLUE formulation of these tasks, which may differ from the actual number of examples – for example in case of multiple-choice questions. Some tasks do not offer a reliable amount of training data and some tasks contain a large number of examples longer than the length limit of our language models.

| Task | Layer | | | | | | | | | | | | Regression |
|-------|-------|-------|-------|-------|-------|-------|-------|------|------|------|------|-------|------------|
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | slope |
| POS | 27.49 | 12.55 | 7.54 | 5.42 | 5.78 | 4.83 | 5.12 | 5.59 | 6.02 | 5.43 | 4.40 | 9.84 | -0.98 |
| DP | 14.65 | 10.78 | 10.99 | 12.66 | 9.92 | 7.47 | 8.00 | 6.04 | 5.66 | 4.58 | 3.89 | 5.35 | -0.89 |
| SRL | 19.38 | 13.70 | 9.80 | 9.56 | 8.44 | 7.04 | 6.99 | 5.61 | 4.85 | 3.83 | 2.70 | 8.11 | -1.04 |
| NER | 18.16 | 9.12 | 6.58 | 4.83 | 6.86 | 6.75 | 6.87 | 6.28 | 6.62 | 4.93 | 5.87 | 17.13 | -0.16 |
| COREF | 7.24 | 9.12 | 7.78 | 10.50 | 11.89 | 12.85 | 12.35 | 8.96 | 5.20 | 4.27 | 3.56 | 6.29 | -0.42 |

Table 5: The per-layer contributions to different edge probing tasks, taken from the layer-wise convex weights γ_k (rendered in percent). To summarize the individual scores, we fit a linear regression line and show its slope in the last column. A negative slope implies stronger representation in the lower layers and vice versa.

It is unclear how to separate the contribution of each layer from the the previous layers here: ℓ_k contains both the previous scaled ℓ_{k-1} and its transformation from a_k and ff_k . On the other hand, our NormFormer-like architecture (Section 4) defines each layer ℓ_k as:

$$\hat{a}_k(x) = x + \text{LN}(a_k(x))$$

$$\ell_k = \hat{a}_k(\ell_{k-1}) + ff_k(\hat{a}_k(\ell_{k-1}))$$

Then it is trivial to calculate the contribution of each layer as $s_k = \ell_k - \ell_{k-1}$. We use this s_k entities to compute the learnt convex combination of all layers $\hat{s} = \sum_{k=1}^{12} \gamma_k s_k$.

Interpreting γ_k as the amount of ‘knowledge’ of a particular task in layer k , we see that POS information is contained primarily in the lowest layers, followed by SRL and DP. On the other hand, NER and CR are represented more strongly in the higher layers, which confirms the related findings in the literature (Rogers et al., 2020).

D Fine-grained results

To ease the evaluation of any future language models trained on BNC, we provide detailed results of all evaluated models in the following tables: GLUE results are shown in Table 6, edge probing performance is given in Table 7 and the BLiMP accuracies in Table 8.

(Super)GLUE. In total, we fine-tune all models on these 14 (Super)GLUE datasets:

- **Boolean Questions** (BoolQ; Clark et al., 2019), a yes/no question answering dataset evaluated with accuracy.
- **The CommitmentBank** (CB; de Marneffe et al., 2019), evaluated with both accuracy and F_1 -score, where the multi-class F_1 is computed as the unweighted average of the F_1 per class.
- **Corpus of Linguistic Acceptability** (CoLA; Warstadt et al., 2019) evaluated with the Matthews correlation coefficient (MCC; Matthews, 1975).
- **Choice of Plausible Alternatives** (COPA; Roemmele et al., 2011), evaluated with accuracy.
- **The Multi-Genre Natural Language Inference Corpus** (MNLI; Williams et al., 2018). Its development set consists of two parts: *matched*, sampled from the same data source as the training set, and *mismatched*, which is sampled from a different domain. Both parts are evaluated with accuracy.
- **The Microsoft Research Paraphrase Corpus** (MRPC; Dolan and Brockett, 2005), eval-

uated with both accuracy and F_1 -score.

- **Multi-Sentence Reading Comprehension** (MultiRC; Khashabi et al., 2018), a multiple choice question answering dataset, evaluated with the exact match accuracy (EM) and F_1 -score (over all answer options).
- **Question-answering Natural Language Inference** (QNLI) constructed from the Stanford Question Answering Dataset (SQuAD; Rajpurkar et al., 2016), evaluated with accuracy.
- **The Quora Question Pairs** (QQP),¹⁴ evaluated with both accuracy and F_1 -score.
- **The Stanford Sentiment Treebank** (SST-2; Socher et al., 2013), evaluated with accuracy.
- **The Semantic Textual Similarity Benchmark** (STS-B; Cer et al., 2017), evaluate with Pearson and Spearman correlation coefficients.
- **Reading Comprehension with Commonsense Reasoning Dataset** (ReCoRD; Zhang et al., 2018), a question answering dataset evaluated with the exact match accuracy (EM) and token-level F_1 -score (maximum over all correct mentions).
- **The Recognizing Textual Entailment datasets** (RTE; Dagan et al., 2006; Bar-Haim et al., 2006; Giampiccolo et al., 2007; Bentivogli et al., 2009), evaluated with accuracy.
- **The Word-in-Context dataset** (WiC; Pilehvar and Camacho-Collados, 2019), evaluated simply with accuracy.

Edge probing. We report the results on part-of-speech tagging (POS), semantic role labeling (SRL), named entity recognition (NER) and coreference resolution (CR) using the annotations from the English part of OntoNotes 5.0 (Weischedel, Ralph et al., 2013). In addition, to further measure the syntactic abilities, we test the dependency parsing (DP) accuracy on the English Web Treebank v2.9 dataset from the Universal Dependencies (Silveira et al., 2014).¹⁵ These choices follow the original work by Tenney et al. (2019), but we do not evaluate on constituency parsing, because the

¹⁴<https://quoradata.quora.com/First-Quora-Dataset-Release-Question-Pairs>

¹⁵Available online at https://github.com/UniversalDependencies/UD_English-EWT.

results suffered from large variation. Instead, we test the syntactic knowledge with DP, which turned out to be more reliable as its variation is negligible (Table 7).

BLiMP. The Benchmark of Linguistic Minimal Pairs for English (Warstadt et al., 2020) consists of 67 tasks. Each focuses on a specific linguistic feature, which is tested with 1 000 automatically generated sentence pairs. Warstadt et al. (2020) clusters these tasks into the following subgroups:

- **Anaphor agreement** tests whether the reflexive pronouns agree with their antecedents.
- **Argument structure** – do verbs appear with the correct types of arguments?
- **Binding** evaluates the correctness of structural relationship between a pronoun and its antecedent.
- **Control/raising** tests syntactic and semantic differences between predicates that embed an infinitival verb predicate.
- **Determiner-noun agreement** checks number agreement between determiners the associated noun.
- **Ellipsis** – can we omit an expression from a sentence?
- **Filler-gap** tests dependencies created by phrasal movement.
- **Irregular forms** checks the correctness of irregular morphology on English past participles.
- **Island effects** – correctness of a possible gap in a filler-gap dependency.
- **NPI licensing** – are the negative polarity items used correctly (e.g. in negation)?
- **Quantifiers** tests the usage of quantifiers.
- **Subject-verb agreement** checks the number agreement between present tense verbs and subjects.

E Hyperparameters

All hyperparameters used to pre-trained and fine-tune our models are listed below: pre-training hyperparameters in Table 9, the GLUE and SuperGLUE fine-tuning hyperparameters in Table 10 and the edge probing hyperparameters in Table 11. BLiMP does not require any special hyperparameters, it need only out-of-the-box predictions of a

pre-trained language model. Note that we will also release the full PyTorch (Paszke et al., 2019) source code, tokenizer and the pre-trained language models in the camera-ready version. Additionally, we will also provide all necessary wrappers for a simple use of our models with the `transformers` library (Wolf et al., 2020).

The training was performed on 128 AMD MI250X GPUs (distributed over 16 compute nodes) and took approximately 8 hours per model in a mixed precision mode. In total, our models consist of 98M parameters; a slightly lower value than BERT’s 110M parameters due to the smaller vocabulary size.

F Wikipedia + BookCorpus dataset replication

The information about the exact Wikipedia dump used for training BERT is unknown and the BookCorpus dataset (Zhu et al., 2015) is no longer available. On top of that, the preprocessing choices are also not known. Our 100M Wikipedia + BookCorpus dataset is thus different from the original BERT pre-training corpus.

We downloaded a fresh English Wikipedia dump from <https://dumps.wikimedia.org/enwiki/20220801/enwiki-20220801-pages-articles-multistream.xml.bz2>, extracted the raw text with WikiExtractor (Attardi, 2015) and segmented each article into sentences with `spaCy`.¹⁶

A replicated version of BookCorpus was obtained from https://the-eye.eu/public/AI/pile_preliminary_components/books1.tar.gz and every book was also segmented with `spaCy`.

After that, the random 100M subset was created by sampling random documents from the full Wikipedia + BookCorpus dataset until the subset contained as many characters as BNC.

G Tokenizer definition

We use the HuggingFace’s `tokenizers` library,¹⁷ to define and train a subword tokenizer on BNC (training split).¹⁸

Following the suggestion of Gowda and May (2020), we set the vocabulary size so that at least 95% of tokens appear more than 100 times. In our case, with the size of $2^{14} = 16\,384$, 95% of

tokens appear more than 166 times in the training split. Their finding comes from the realm of neural machine translation, we have not evaluated how it aligns with language modeling. Nevertheless, we believe that a comparative study of different tokenizer settings makes an interesting future work; we suspect that the effects will be more pronounced with BNC, due to its limited size.

¹⁶ <https://spacy.io/>

¹⁷ <https://huggingface.co/tokenizers/>

¹⁸ We share the full definition of the tokenizer in censored-for-review.com.

| Task | Metric | BERT (100M subset) | MLM | | | NSP | | Training steps | | |
|----------------|-----------------|--------------------|--------------------|--------------------|--------------------|--------------------|--------------------|--------------------|--------------------|--------------------|
| | | | subword | word | span | document | order | 2× | 0.5× | 0.25× |
| BoolQ | accuracy | 75.16±0.48 | 74.87±0.26 | 75.94±0.16 | 75.08±0.94 | 74.75±0.71 | 74.80±1.07 | 74.87±0.62 | 74.84±0.71 | 74.08±0.56 |
| CB | accuracy | 78.93±3.43 | 76.06±2.40 | 84.64±3.48 | 75.71±2.71 | 82.86±1.60 | 83.57±1.49 | 80.00±1.96 | 74.28±3.91 | 77.14±3.44 |
| | F ₁ | 72.11±6.73 | 72.78±5.17 | 80.42±4.52 | 71.91±8.36 | 77.78±2.31 | 80.99±3.10 | 72.56±4.04 | 66.73±4.23 | 80.69±3.45 |
| CoLA | MCC | 59.36±0.96 | 57.17±1.92 | 58.28±0.59 | 58.69±1.43 | 59.73±1.34 | 57.91±1.51 | 57.47±1.62 | 59.98±1.40 | 58.30±1.15 |
| COPA | accuracy | 60.40±5.03 | 59.20±2.28 | 64.00±5.43 | 59.40±5.03 | 72.00±1.87 | 62.80±2.77 | 54.20±1.96 | 58.00±3.32 | 61.60±2.19 |
| MNLI | matched acc. | 84.22±0.12 | 85.14±0.16 | 84.93±0.21 | 85.05±0.19 | 85.21±0.25 | 84.72±0.15 | 85.17±0.16 | 84.40±0.29 | 83.82±0.16 |
| | mismatched acc. | 84.00±0.05 | 84.78±0.17 | 85.05±0.13 | 85.35±0.15 | 85.36±0.21 | 84.73±0.19 | 85.29±0.14 | 84.60±0.16 | 83.71±0.16 |
| | HANS acc. | 62.47±1.68 | 64.39±1.28 | 63.75±0.76 | 65.60±0.53 | 60.50±1.24 | 64.16±1.86 | 65.32±1.14 | 62.35±0.82 | 58.63±1.35 |
| MRPC | accuracy | 84.31±0.71 | 85.00±0.94 | 85.54±0.90 | 87.45±0.86 | 86.52±0.81 | 85.93±0.59 | 86.47±0.80 | 86.27±1.12 | 85.29±0.83 |
| | F ₁ | 89.06±0.48 | 89.51±0.64 | 89.83±0.61 | 91.20±0.62 | 90.39±0.59 | 89.99±0.48 | 90.54±0.61 | 90.39±0.73 | 89.57±0.63 |
| MultiRC | F ₁ | 67.25±0.57 | 67.61±0.86 | 68.10±0.85 | 71.93±0.73 | 71.90±0.35 | 71.91±0.35 | 66.45±2.12 | 67.30±0.62 | 65.02±1.00 |
| | exact match | 18.51±0.88 | 19.58±1.51 | 18.76±1.54 | 25.25±1.37 | 24.91±0.40 | 27.63±0.83 | 17.19±2.70 | 18.65±0.44 | 16.66±0.77 |
| QNLI | accuracy | 90.80±0.25 | 90.00±0.25 | 90.57±0.29 | 91.46±0.20 | 90.32±0.18 | 90.36±0.25 | 90.33±0.27 | 90.36±0.16 | 89.08±0.24 |
| QPP | accuracy | 91.01±0.05 | 90.94±0.06 | 90.85±0.07 | 91.01±0.10 | 91.00±0.14 | 90.90±0.08 | 91.01±0.08 | 90.77±0.04 | 90.51±0.09 |
| | F ₁ | 87.85±0.07 | 87.81±0.08 | 87.73±0.07 | 87.87±0.14 | 87.94±0.19 | 87.76±0.10 | 87.92±0.13 | 87.57±0.05 | 87.24±0.13 |
| SST-2 | accuracy | 92.06±0.48 | 92.71±0.40 | 92.71±0.24 | 92.80±0.50 | 92.18±0.38 | 92.11±0.25 | 92.34±0.59 | 92.82±0.40 | 91.67±0.37 |
| STS-B | Pearson corr. | 86.34±0.29 | 87.44±0.33 | 87.53±0.19 | 87.99±0.11 | 89.50±0.14 | 89.11±0.25 | 87.83±0.19 | 86.93±0.50 | 85.80±0.18 |
| | Spearman corr. | 86.10±0.31 | 87.24±0.32 | 87.45±0.20 | 87.72±0.10 | 89.06±0.12 | 88.82±0.22 | 87.67±0.21 | 86.73±0.47 | 85.54±0.20 |
| ReCoRD | F ₁ | 65.48±0.64 | 63.15±3.19 | 68.36±1.59 | 70.71±1.81 | 66.51±0.33 | 67.73±1.00 | 62.93±3.12 | 64.68±1.90 | 57.59±2.06 |
| | exact match | 64.81±0.62 | 62.48±3.19 | 67.61±1.58 | 70.03±1.78 | 65.84±0.33 | 67.04±1.02 | 62.26±3.08 | 63.93±1.89 | 56.88±2.07 |
| RTE | accuracy | 62.38±3.00 | 60.65±1.92 | 60.51±2.07 | 60.51±2.61 | 66.50±1.12 | 69.68±1.28 | 58.34±2.56 | 56.82±1.63 | 58.19±0.59 |
| WiC | accuracy | 66.36±1.59 | 66.46±1.21 | 67.40±0.43 | 69.18±1.04 | 70.78±0.94 | 68.90±0.60 | 67.52±1.35 | 66.71±0.99 | 68.46±0.71 |
| Average | | 74.04 ±2.20 | 73.63 ±1.75 | 75.20 ±1.99 | 75.12 ±2.39 | 76.69 ±0.96 | 76.21 ±1.22 | 73.34 ±1.91 | 73.39 ±1.67 | 73.15 ±1.33 |

Table 6: Detailed development GLUE and SuperGLUE results for all tested models. We show the mean and standard deviation statistics over 5 runs with different random seeds (changed only for fine-tuning, the pre-trained models are kept the same).

| Model | | POS | DP | SRL | NER | CR | Average |
|-----------------------|--------------------|------------|------------|------------|------------|------------|------------|
| BERT (100M subset) | | 97.94±0.01 | 95.03±0.04 | 92.34±0.06 | 95.91±0.12 | 95.27±0.10 | 95.30±0.08 |
| MLM | subword | 97.91±0.01 | 94.99±0.02 | 92.44±0.03 | 95.77±0.06 | 95.30±0.07 | 95.28±0.05 |
| | whole-word | 97.90±0.01 | 94.99±0.05 | 92.42±0.08 | 95.71±0.07 | 95.64±0.07 | 95.33±0.06 |
| | span | 97.91±0.01 | 94.80±0.03 | 92.32±0.02 | 95.56±0.07 | 95.46±0.14 | 95.21±0.07 |
| NSP | subword + document | 97.92±0.01 | 95.01±0.03 | 92.42±0.06 | 95.76±0.07 | 95.25±0.11 | 95.28±0.07 |
| | subword + order | 97.85±0.01 | 94.92±0.06 | 92.25±0.07 | 95.22±0.05 | 95.25±0.11 | 95.10±0.07 |
| Steps | subword + 2× | 97.93±0.01 | 94.95±0.10 | 92.47±0.03 | 95.63±0.11 | 95.58±0.04 | 95.31±0.07 |
| | subword + 1/2× | 97.90±0.02 | 95.02±0.04 | 92.38±0.05 | 95.46±0.03 | 95.43±0.05 | 95.24±0.04 |
| | subword + 1/4× | 97.88±0.01 | 94.81±0.07 | 92.21±0.03 | 95.32±0.08 | 95.00±0.18 | 95.04±0.10 |
| Random initialization | | 69.85±0.42 | 66.25±0.20 | 70.87±0.21 | 73.16±0.60 | 85.56±0.46 | 73.14±0.41 |

Table 7: Detailed edge probing results for all tested models. † The BERT_{base} scores in the first row are taken from Tenney et al. (2019). The last row shows the edge probing results with a randomly initialized language model – its performance hints at how much information is included in the probes themselves.

| BLiMP subgroups | BERT (100M subset) | MLM | | | NSP | | Size | | |
|---------------------------|--------------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|
| | | subword | word | span | document | order | medium | small | tiny |
| Anaphor agreement | 93.20 | 93.95 | 92.65 | 94.50 | 93.00 | 94.00 | 94.65 | 93.30 | 94.60 |
| Argument structure | 78.95 | 80.73 | 67.93 | 80.98 | 81.58 | 80.61 | 81.54 | 80.98 | 81.99 |
| Binding | 77.04 | 78.34 | 74.60 | 77.26 | 77.74 | 76.60 | 77.33 | 76.43 | 77.03 |
| Control/raising | 73.76 | 79.68 | 79.90 | 81.02 | 78.18 | 78.32 | 78.84 | 79.80 | 78.82 |
| Determiner-noun agreement | 95.91 | 96.74 | 93.48 | 97.45 | 97.09 | 96.26 | 97.09 | 96.73 | 96.96 |
| Ellipsis | 88.25 | 88.10 | 85.55 | 90.95 | 88.65 | 86.70 | 90.25 | 87.70 | 88.70 |
| Filler-gap | 85.44 | 83.87 | 83.30 | 85.86 | 87.23 | 83.46 | 85.20 | 84.73 | 84.20 |
| Irregular forms | 88.45 | 91.45 | 86.75 | 94.40 | 88.30 | 86.70 | 92.35 | 92.65 | 93.10 |
| Island effects | 70.91 | 74.99 | 76.71 | 74.34 | 73.98 | 74.62 | 72.14 | 74.86 | 72.34 |
| NPI licensing | 81.07 | 82.40 | 81.73 | 82.36 | 83.24 | 78.79 | 82.43 | 84.96 | 82.86 |
| Quantifiers | 69.98 | 68.88 | 70.00 | 74.13 | 64.50 | 68.77 | 72.58 | 67.10 | 67.53 |
| Subject-verb agreement | 91.78 | 92.64 | 83.97 | 92.92 | 92.13 | 90.00 | 91.72 | 92.25 | 92.22 |
| Accuracy | 81.95 | 83.42 | 80.05 | 84.18 | 83.31 | 82.17 | 83.45 | 83.47 | 83.15 |

Table 8: Detailed BLiMP results for all tested models.

| Hyperparameter | Base |
|------------------------|--|
| Number of layers | 12 |
| Hidden size | 768 |
| FF intermediate size | 2 048 |
| Vocabulary size | 16 384 |
| FF activation function | GEGLU |
| Attention heads | 12 |
| Attention head size | 64 |
| Dropout | 0.1 |
| Attention dropout | 0.1 |
| Training steps | 31 250 |
| Batch size | 32 768 (90% steps) / 8 192 (10% steps) |
| Sequence length | 128 (90% steps) / 512 (10% steps) |
| Tokens per step | 4 194 304 |
| Warmup steps | 500 (1.6% steps) |
| Initial learning rate | 0.01 |
| Final learning rate | 0.001 |
| Learning rate decay | cosine |
| Weight decay | 0.1 |
| Layer norm ϵ | 1e-5 |
| Optimizer | LAMB |
| LAMB ϵ | 1e-6 |
| LAMB β_1 | 0.9 |
| LAMB β_2 | 0.98 |
| Gradient clipping | 2.0 |

Table 9: Pre-training hyperparameters. The models differ only in their hidden size and number of layers, the learning rate schedule and other training settings are kept identical.

| Hyperparameter | ReCoRD | MNLI, QQP, QNLI | BoolQ, CoLA, COPA, SST-2, MultiRC, MRPC, STSB | | | CB |
|---------------------|--------|-----------------|---|--------|--------|----|
| | | | RTE, WiC | | | |
| Batch size | 32 | 32 | 32 | 16 | 8 | |
| Number of epochs | 1 | 4 | 8 | 8 | 16 | |
| Dropout | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | |
| Warmup steps | 10% | 10% | 10% | 10% | 10% | |
| Peak learning rate | 3e-5 | 3e-5 | 3e-5 | 3e-5 | 3e-5 | |
| Learning rate decay | linear | linear | linear | linear | linear | |
| Weight decay | 0.01 | 0.01 | 0.01 | 0.01 | 0.01 | |
| Optimizer | AdamW | AdamW | AdamW | AdamW | AdamW | |
| Adam ϵ | 1e-6 | 1e-6 | 1e-6 | 1e-6 | 1e-6 | |
| Adam β_1 | 0.9 | 0.9 | 0.9 | 0.9 | 0.9 | |
| Adam β_2 | 0.999 | 0.999 | 0.999 | 0.999 | 0.999 | |

Table 10: Hyperparameters for fine-tuning the GLUE and SuperGLUE tasks. We use the same hyperparameters for all models, not performing any per-model hyperparameter search.

| Hyperparameter | POS, SRL, NER, CR | DP |
|-------------------------|-------------------|--------|
| Batch size | 128 | 128 |
| Number of epochs | 5 | 10 |
| Dropout | 0.25 | 0.25 |
| Downsampled hidden size | 256 | 256 |
| Attention pooling heads | 4 | 4 |
| MLP hidden layers | 1 | 1 |
| Starting learning rate | 6e-3 | 6e-3 |
| Learning rate decay | cosine | cosine |
| Weight decay | 0.01 | 0.01 |
| Optimizer | AdamW | AdamW |
| Adam ϵ | 1e-6 | 1e-6 |
| Adam β_1 | 0.9 | 0.9 |
| Adam β_2 | 0.999 | 0.999 |
| Gradient clipping | 2.0 | 2.0 |

Table 11: Edge probing hyperparameters.

```

1 # Oral history project: interview
2
3 Britta: 'Can you tell us what er what section you work in?'
4
5 Eliazar: 'I work at the weaving'
6
7 Britta: 'In the weaving?'
8
9 Eliazar: 'section, aha.
10 And'
11
12 Britta: 'And what do you do?'
13
14 Eliazar: 'I'm what you call a Axminster handler'
15
16 Britta: 'Aha.'
17
18 Eliazar: 'which involves like when the frames comes off the weaving and they're yarn left, I strip the
↳ yarn off.'
19
20 Britta: 'Mhm.'
21
22 Eliazar: 'Off the, the, the weaving frames.'
23
24 Britta: 'Mhm.'
25
26 Eliazar: 'That's basically my, aha.'
27
28 Britta: 'It's quite spec specialized so'
29
30 Eliazar: 'No no, no, no.
31 It's not specialized, no.'
32
33 Britta: 'Mhm, have you ever worked in any other factory?'
34
35 Eliazar: 'Aha, I worked in spooling, I've been left now two year.'
36
37 Britta: 'And how did you find that?'
38
39 Eliazar: 'Er, I liked the spooling but some I just don't know, some of the girls get kind of one [UNK]
↳ one thing by the other I can object to, I think it was actually the atmosphere of the, the girls
↳ that worked in the department that I'

```

Listing 1: A random example of the first few lines from a preprocessed spoken document from BNC. Notice that the text is divided into speech turns (paragraphs), each starting with the name of a speaker. Line 39 contains a special [UNK] token in place of an incomprehensible word or phrase.

```

1 # Organizing knowledge: an introduction to information retrieval
2
3 ## SUBJECTS
4
5 ### The subject approach: introduction, processes, tools and simple evaluation
6
7 ##### 1.2.1 Subjects
8
9 Users often approach information sources not with names (as have been considered in Part II), but with
↳ a question that requires an answer or a topic for study.
10 Users seek documents or information concerned with a particular subject.
11 In order to make some provision for this common approach to information sources, it is necessary to
↳ arrange documents- and document surrogates in catalogues, indexes bibliographies, computer
↳ databases and so on - in such a way that items on specific subjects can be retrieved.
12 Thus, the subject approach is extremely important in the access to and the exploitation of information,
↳ documents and data.
13
14 Before we discuss the provision that libraries and information workers make for the subject approach,
↳ it may be useful to consider the preliminary question: What is a subject?
15 In talking about a subject we generally refer to a given area of knowledge or to the contents of an
↳ information source of a given scope.
16 A subject might be considered to be defined by:
17
18 - an area of interest,
19
20 - an area in which an individual researcher or professional works,
21
22 - an area in which an individual writes or an area of knowledge which is studied.

```

Listing 2: A sample of the first few lines from a written BNC article. Note the H1-level header with the title of the whole document and then the title of a chapter, section and subsection in the lines below. This sample also contains a special text block with a list in the last lines.