Pretrained Language Model Embryology: The Birth of ALBERT

Cheng-Han Chiang
National Taiwan University, Taiwan
dcml0714@gmail.com

Sung-Feng Huang
National Taiwan University, Taiwan
f06942045@ntu.edu.tw

Hung-yi Lee
National Taiwan University, Taiwan
hungyilee@ntu.edu.tw

Abstract

While behaviors of pretrained language models (LMs) have been thoroughly examined, what happened during pretraining is rarely studied. We thus investigate the developmental process from a set of randomly initialized parameters to a totipotent[1] language model, which we refer to as the *embryology* of a pretrained language model. Our results show that ALBERT learns to reconstruct and predict tokens of different parts of speech (POS) in different learning speeds during pretraining. We also find that linguistic knowledge and world knowledge do not generally improve as pretraining proceeds, nor do downstream tasks' performance. These findings suggest that knowledge of a pretrained model varies during pretraining, and having more pretrain steps does not necessarily provide a model with more comprehensive knowledge. We provide source codes and pretrained models to reproduce our results at https://github.com/d223302/albert-embryology.

1 Introduction

The world of NLP has gone through some tremendous revolution since the proposal of contextualized word embeddings. Some big names are ELMo (Peters et al., 2018), GPT (Radford et al.), and BERT (Devlin et al., 2019), along with its variants (Sanh et al., 2019; Liu et al., 2019b; Lan et al., 2019). Performance boosts on miscellaneous downstream tasks have been reported by finetuning these totipotent pretrained language models. With a view to better grasping what has been learned by these contextualized word embedding models, probing is generally applied to the pretrained models and the models finetuned from them. Probing targets can range from linguistic knowledge, including semantic roles and syntactic structures (Liu et al., 2019a; Tenney et al., 2019, 2018; Hewitt and Manning, 2019), to world knowledge (Petroni et al., 2019).

While the previous work focuses on what knowledge has been learned after pretraining of transformer-based language models, few delve into their dynamics during pretraining. What happened during the training process of a deep neural network model has been widely studied, including Gur-Ari et al. (2018), Frankle et al. (2019), Raghu et al. (2017), Morcos et al. (2018). Some previous works also study the dynamics of the training process of an LSTM language model (Saphra and Lopez, 2018, 2019), but the training dynamics of a large scale pretrained language models are not well-studied. In this work, we probe ALBERT (Lan et al., 2019) during its pretraining phase every N parameter update steps and study what it has learned and what it can achieve so far. We perform a series of experiments, detailed in the following sections, to investigate the development of predicting and reconstructing tokens (Section 3), how linguistic and world knowledge evolve through time (Section 4, Section 6), and whether amassing those information serves as an assurance of good downstream task performances (Section 5).

We have the following findings based on ALBERT:

- The prediction and reconstruction of tokens with different POS tags have different learning speeds. (Section 3)
- Semantic and syntactic knowledge is developed simultaneously in ALBERT. (Section 4)
- Finetuning from model pretrained for 250k steps gives a decent GLUE score (80.23), and

[1] According to Wikipedia, totipotency is the ability of a single cell to divide and produce all of the differentiated cells in an organism. We use its adjective form here to refer to the ability of a pretrained model which can be finetuned for a variety of downstream tasks.

Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, pages 6813–6828,
November 16–20, 2020.

further pretrain steps only make the GLUE score rise as high as 81.50.

- While ALBERT does generally gain more world knowledge as pretraining goes on, the model seems to be dynamically renewing its knowledge about the world. (Section 6)

While we only include the detailed results of ALBERT in the main text, we find that the results also generalize to the other two transformer-based language models, ELECTRA (Clark et al., 2019) and BERT, which are quite different from ALBERT in the sense of pretext task and model architecture. We put the detailed results of ELECTRA and BERT in the appendix.

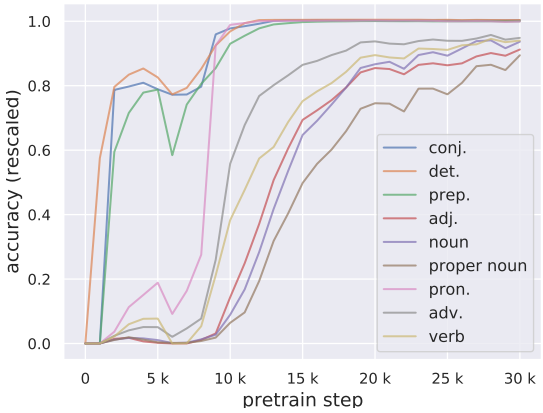
2 Pretraining ALBERT

ALBERT is a variant of BERT with cross-layer parameters sharing and factorized embedding parameterization. The reason why we initially chose ALBERT as our subject lies in its parameter efficiency, which becomes a significant issue when we need to store 1000 checkpoints during the pretraining process.

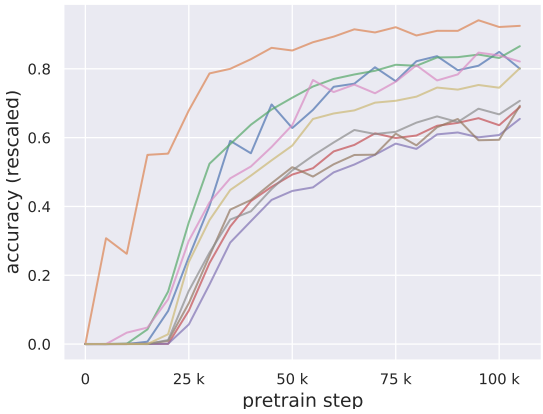
To investigate what happened during the pretraining process of ALBERT, we pretrained an ALBERT-base model ourselves. To maximally reproduce the results in Lan et al. (2019), we follow most of the training hyperparameters in the original work, only modifying some hyperparameters to fit in our limited computation resources[2]. We also follow Lan et al. (2019), using English Wikipedia as our pretraining data, and we use the Project Gutenberg Dataset (Lahiri, 2014) instead of BookCorpus. The total size of the corpus used in pretraining is 16GB. The pretraining was done on a single Cloud TPU V3 and took eight days to finish 1M pretrain steps, costing around 700 USD. More details on pretraining are specified in appendix B.1.

3 Learning to Predict the Masked Tokens and Reconstruct the Input Tokens

During the pretraining stage of a masked LM (MLM), it learns to predict masked tokens based on the remaining unmasked part of the sentence, and it also learns to reconstruct token identities of unmasked tokens from their output representations of the model. Better prediction and reconstruction results indicate the model being able to utilize contextual information. To maximally reconstruct the input tokens, the output representations must keep sufficient information regarding token identities.



(a) Token reconstruction

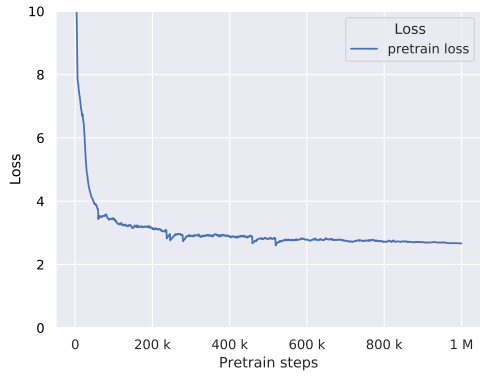


(b) Mask prediction

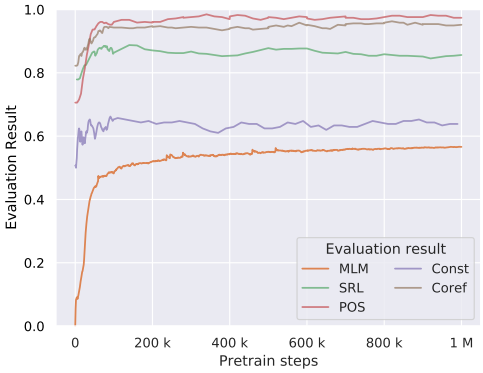
Figure 1: Rescaled accuracy of token reconstruction and mask prediction during pretraining. We rescale the accuracy of each line by the accuracy when the model is fully pretrained, i.e., the accuracy after pretraining 1M steps. Token reconstruction are evaluated every 1K pretrain steps, and mask prediction evaluated every 5K steps.

We investigate the behavior of mask prediction and token reconstruction for tokens of different POS during the early stage of pretraining. We use the POS tagging in OntoNotes 5.0 (Weischedel et al., 2013) in this experiment. For the mask prediction part, we mask a whole word (which may contain multiple tokens) of an input sentence, feed the masked sentence into ALBERT, and predict the masked token(s). We evaluate the prediction performance by calculating the prediction's accuracy based on POS of the word; the predicted token(s) should exactly match the original token(s) to be deemed an accurate prediction. As for the token reconstruction part, the input to the model is simply

[2]We use the official implementation of ALBERT at https://github.com/google-research/albert.



(a) Total loss during pretraining



(b) Masked LM accuracy and F1 scores of different probing tasks over the course of pretraining

Figure 2: The probing results of hidden representation from layer 8; all four tasks are evaluated with test set of OntoNotes 5.0 and F1 scores are reported. MLM accuracy is also shown. We smoothed the lines by averaging 3 consecutive data points for better illustration. The unsmoothed result is in Appendix D.3.

the original sentence.

The results of reconstruction are shown in Figure 1(a). ALBERT first learns to reconstruct function words, e.g., determiners, prepositions, and then gradually learns to reconstruct content words in the order of verb, adverb, adjective, noun, and proper noun. We also found that different forms and tenses of a verb do not share the same learning schedule, with third-person singular present being the easiest to reconstruct and present participle being the hardest (shown in Appendix C.2). The prediction results in Figure 1(b) reveal that learning mask prediction is generally more challenging than token reconstruction. ALBERT learns to predict masked tokens with an order similar to token reconstruction, though much slower and less accurate. We find that BERT also learns to perform mask prediction and token reconstruction in a similar fashion, with the results provided in Appendix C.4.

4 Probing Linguistic Knowledge Development During Pretraining

Probing is widely used to understand what kind of information is encoded in embeddings of a language model. In short, probing experiments train a task-specific classifier to examine if token embeddings contain the knowledge required for the probing task. Different language models may give different results on different probing tasks, and representations from different layers of a language model may also contain different linguistic information (Liu et al., 2019a; Tenney et al., 2018).

Our probing experiments are modified from the "edge probing" framework in Tenney et al. (2018). Hewitt and Liang (2019) previously showed that probing models should be *selective*, so we use linear classifiers for probing. We select four probing tasks for our experiments: part of speech (POS) tagging, constituent (const) tagging, coreference (coref) resolution, and semantic role labeling (SRL). The former two tasks probe syntactic knowledge hidden in token embeddings, and the last two tasks are designed to inspect the semantic knowledge provided by token embeddings. We use annotations provided in OntoNotes 5.0 (Weischedel et al., 2013) in our experiments.

The probing results are shown in Figure 2b. We observe that all four tasks show similar trends during pretraining, indicating that semantic knowledge and syntactic knowledge are developed simultaneously during pretraining. For syntactically related tasks, the performance of both POS tagging and constituent tagging boost very fast in the first 100k pretrain steps, and no further improvement can be seen throughout the remaining pretraining process, while performance fluctuates from time to time. We also observe an interesting phenomenon: the probed performances of SRL peak at around 150k steps and slightly decay over the remaining pretraining process, suggesting that some information in particular layers related to probing has been dwindling while the ALBERT model strives to advance its performance on the pretraining objective. The loss of the pretraining objective is also shown in Figure 2a.

Scrutinizing the probing results of different layers (Figure 3 and Appendix D.3), we find that the behaviors among different layers are slightly different. While the layers closer to output layer perform worse than layers closer to input layer at the beginning of pretraining, their performances rise

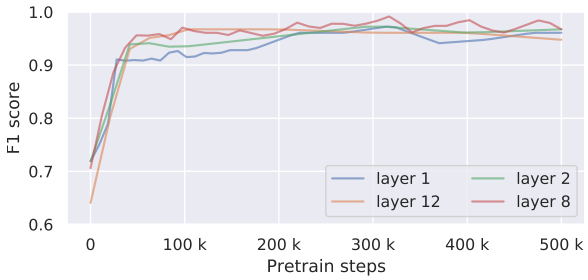


Figure 3: The probing results of POS during pretraining. Layers are indexed from the input layer to the output layer.

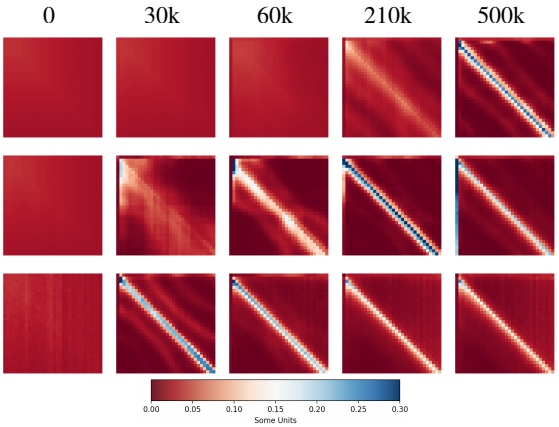


Figure 4: Attention patterns of head 11 across layer 1 (first row), 2 (second row), and 8 (third row) during pretraining. Pretrain steps labeled atop the attention map. We averaged the attention maps of different input sentences to get the attention pattern of a single head.

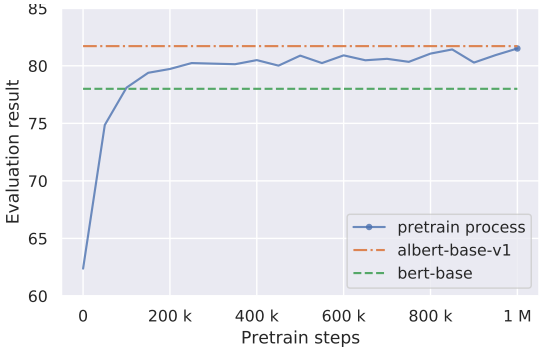
drastically and eventually surpass the top few layers; however, they start to decay after they reach best performances. This implies the last few layers of ALBERT learn faster than the top few layers. This phenomenon is also revealed by observing the attention patterns across different layers during pretraining. Figure 4 shows that the diagonal attention pattern (Kovaleva et al., 2019) of layer 8 emerges earlier than layer 2, with the pattern of layer 1 looms the last[3].

5 Does Expensive and Lengthy Pretraining Guarantee Exceptional Results on Downstream Tasks?

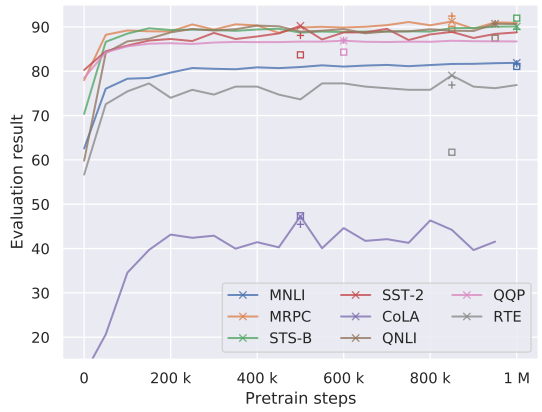
While Devlin et al. (2019) and Lan et al. (2019) have shown that more pretrain steps lead to better

[3]GIF files are provided in this website: https://albertembryo.wordpress.com/

[4]GLUE score of albert-base-v1 and bert-base are obtained by finetuning ALBERT and BERT models from HuggingFace(Wolf et al., 2019)



(a) GLUE scores over pretraining. GLUE scores of albert-base-v1 and bert-base are also shown by horizontal lines.[4].



(b) Performance of individual tasks in GLUE benchmark. Best result during pretraining marked with 'x'. Performances of albert-base-v1 and bert-base-uncased are marked with '+' and square respectively.

Figure 5: Downtream evaluation of ALBERT on development set every 50k pretrain steps. GLUE score is averaged among all tasks except WNLI. Evaluation metrics: MRPC and QQP: F1, STS-B: Spearman corr., others: accuracy. The result of MNLI is the average of matched and mismatched.

GLUE scores, whether the performance gain of downstream tasks is proportional to the resources spent on additional pretrain steps is unknown. This drives us to explore the downstream performance of the ALBERT model before fully pretrained. We choose GLUE benchmark (Wang et al., 2018) for downstream evaluation, while excluding WNLI, following Devlin et al. (2019).

We illustrate our results of the downstream performance of the ALBERT model during pretraining in Figure 5. While the GLUE score gradually increases as pretraining proceeds, the performance after 250k does not pale in comparison with a fully pretrained model (80.23 v.s. 81.50). From Figure 5b, we also observe that most GLUE tasks reach comparable results with their fully pretrained counterpart over 250k pretrain steps, except for

MNLI and QNLI, indicating NLI tasks do benefit from more pretrain steps when the training set is large.

We also finetuned BERT and ELECTRA models as pretraining proceeds, and we observe similar trends. The GLUE scores of the BERT and ELECTRA model rise drastically in the first 100k pretrain steps, and then the performance increments less slowly afterward. We put the detailed result of these two models in Section E.4.

We conclude that it may not be necessary to train an ALBERT model until its pretraining loss converges to obtain exceptional downstream performance. The majority of its capability for downstream tasks has already been learned in the early stage of pretraining. Note that our results do not contradict previous findings in Devlin et al. (2019), Liu et al. (2019b), and Clark et al. (2019), all of which showing that downstream tasks do benefit from more pretrain steps; we show that the performance gain on downstream tasks in latter pretrain steps might be disproportional to the cost on more pretrain steps.

6 World Knowledge Development During Pretraining

Petroni et al. (2019) has reported that language models contain world knowledge. To examine the development of world knowledge of a pretrained language model, we conduct the same experiment as in Petroni et al. (2019). We use a subset of T-REx (Elsahar et al., 2018) from the dataset provided by Petroni et al. (2019) to evaluate ALBERT's world knowledge development.

The results are shown in Figure 6, in which we observe that world knowledge is indeed built up during pretraining, while performance fluctuates occasionally. From Figure 6, it is clear that while some types of knowledge stay static during pretraining, some vary drastically over time, and the result of a fully pretrained model (at 1M steps) may not contain the most amount of world knowledge. We infer that world knowledge of a model depends on the corpus it has seen recently, and it tends to forget some knowledge that it has seen long ago. These results imply that it may not be sufficient to draw a conclusion on ALBERT's potential as a knowledge base merely based on the final pretrained one's behavior. We also provide qualitative results in Appendix F.2.

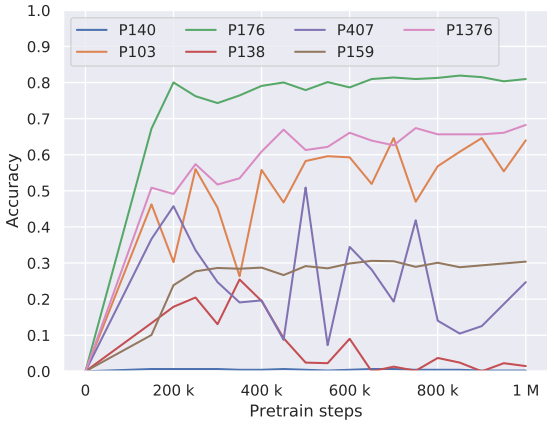


Figure 6: World knowledge development during pretraining evaluated every 50k pretrain steps. Types of relation, and template are shown in Table 1

Type	Query template
P140	[X] is affiliated with the [Y] religion .
P103	The native language of [X] is [Y] .
P176	[X] is produced by [Y] .
P138	[X] is named after [Y] .
P407	[X] was written in [Y] .
P159	The headquarter of [X] is in [Y] .
P1376	[X] is the capital of [Y] .

Table 1: Relations in Figure 6. We fill in [X] with the subject, [Y] with [MASK] and ask model to predict Y.

7 Conclusion

Although finetuning from pretrained language models puts in phenomenal downstream performance, the reason is not fully uncovered. This work aims to unveil the mystery of the pretrained language model by looking into how it evolves. Our findings show that the learning speeds for reconstructing and predicting tokens differ across POS. We find that the model acquires semantic and syntactic knowledge simultaneously at the early pretraining stage. We show that the model is already prepared for finetuning on downstream tasks at its early pretraining stage. Our results also reveal that the model's world knowledge does not stay static even when pretraining loss converges. We hope our work can bring more insights into what makes a pretrained language model a pretrained language model.

Acknowledgements

We thank all the reviewers' valuable suggestions and efforts towards improving our manuscript. This work was supported by Delta Electronics, Inc.

References

Kevin Clark, Minh-Thang Luong, Quoc V Le, and Christopher D Manning. 2019. Electra: Pre-training text encoders as discriminators rather than generators. In *International Conference on Learning Representations*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT (1)*.

Hady Elsahar, Pavlos Vougiouklis, Arslen Remaci, Christophe Gravier, Jonathon Hare, Frederique Laforest, and Elena Simperl. 2018. T-rex: A large scale alignment of natural language with knowledge base triples. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*.

Jonathan Frankle, David J Schwab, and Ari S Morcos. 2019. The early phase of neural network training. In *International Conference on Learning Representations*.

Aaron Gokaslan and Vanya Cohen. 2019. Openwebtext corpus. http://Skylion007.github.io/OpenWebTextCorpus.

Guy Gur-Ari, Daniel A Roberts, and Ethan Dyer. 2018. Gradient descent happens in a tiny subspace. *arXiv preprint arXiv:1812.04754*.

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)*, pages 2733–2743.

John Hewitt and Christopher D Manning. 2019. A structural probe for finding syntax in word 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 4129–4138.

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.

Shibamouli Lahiri. 2014. Complexity of Word Collocation Networks: A Preliminary Structural Analysis. In *Proceedings of the Student Research Workshop at the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 96–105, Gothenburg, Sweden. Association for Computational Linguistics.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. Albert: A lite bert for self-supervised learning of language representations. In *International Conference on Learning Representations*.

Nelson F Liu, Matt Gardner, Yonatan Belinkov, Matthew E Peters, and Noah A Smith. 2019a. 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, Volume 1 (Long and Short Papers)*, pages 1073–1094.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019b. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Ari Morcos, Maithra Raghu, and Samy Bengio. 2018. Insights on representational similarity in neural networks with canonical correlation. In *Advances in Neural Information Processing Systems*, pages 5727–5736.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proc. of NAACL*.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? 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 2463–2473.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners.

Maithra Raghu, Justin Gilmer, Jason Yosinski, and Jascha Sohl-Dickstein. 2017. Svcca: Singular vector canonical correlation analysis for deep learning dynamics and interpretability. In *Advances in Neural Information Processing Systems*, pages 6076–6085.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for squad. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 784–789.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Naomi Saphra and Adam Lopez. 2018. Language models learn pos first. In *Proceedings of the 2018*

EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP, pages 328–330.

Naomi Saphra and Adam Lopez. 2019. Understanding learning dynamics of language models with svcca. 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 3257–3267.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. Bert rediscovers the classical nlp pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4593–4601.

Ian Tenney, Patrick Xia, Berlin Chen, Alex Wang, Adam Poliak, R Thomas McCoy, Najoung Kim, Benjamin Van Durme, Samuel R Bowman, Dipanjan Das, et al. 2018. What do you learn from context? probing for sentence structure in contextualized word representations. In *International Conference on Learning Representations*.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355.

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.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, R'emi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771.

A Modifications from the Reviewed Version

We made some modifications in the camera-ready version, mostly based on the reviewers' recommendations and for better reproducibility.

- We add the result of BERT and ELECTRA in Section 3, Section 4, and Section 5.
- We reimplement the source code for Section 4 and renew the experiment results accordingly. While the exact values are slightly different, the general trends are the same and do not affect our observation.
- We add the results of coreference resolution in our probing experiments, following the reviewers' suggestion.
- We polish our wordings and presentations in text and figures.

B Pretraining

B.1 ALBERT

As mentioned in the main text, we only modified a few hyperparameters to fit in out limited computation resources, listed in Table 2. The Wikipedia corpus used in our pretraining can be download from `https://dumps.wikimedia.org/enwiki/latest/enwiki-latest-pages-articles.xml.bz2`, and the Gutenburg dataset can be download from `https://web.eecs.umich.edu/~lahiri/gutenberg_dataset.html`. The number of parameters in our ALBERT model is 12M.

Batch size	512
Learning rate	6.222539674E-4
Total steps	1M
Warmup steps	25k

Table 2: Pretraining hyperparemeters for ALBERT.

B.2 BERT

We use the same dataset as we trained ALBERT to pretrain BERT. We pretrained a BERT-base-uncased model using the official implementation of BERT at `https://github.com/google-research/bert`, and we follow all hyperparameters of the original implementation. Note that the Devlin et al. (2019) mentioned they trained BERT with a maximum sequence length of 128 for the first 900K steps, and then trained the model with a maximum sequence length 512 for the rest 100K steps; we follow this training procedure. The number of parameters in our BERT model is 110M.

B.3 ELECTRA

We use OpenWebTextCorpus (Gokaslan and Cohen, 2019) from `https://skylion007.github.io/OpenWebTextCorpus/` to pretrain an Electra-base model. We pretrained this model using the official implementation of ELECTRA at `https://github.com/google-research/electra`, and we follow all hyperparameters of the original implementation. The number of parameters in our ELECTRA model used for finetuning (the discriminator part) is 110M.

C Mask Predict and Token Reconstruction

C.1 Dataset

As mentioned in Section 3, we use the POS annotations in OntoNotes 5.0, and we only use the CoNLL-2012 test set for our experiments. While there are 48 POS labels, we only report the mask prediction and token reconstruction of a much smaller subset—those we are more familiar with. The statistics of these POS are in Table 3.

POS	Count
Conjunction	5109
Determiner	14763
Preposition	18059
Adjective	9710
Adverb	7992
Verb (all forms)	21405
Noun	29544
Proper noun	13144

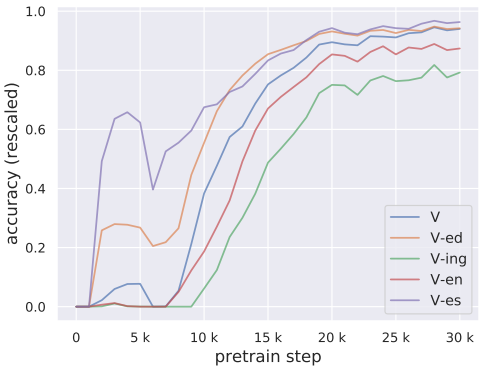
Table 3: Statistics of POS used in experiments.

Verb form	Count
Base form	5865
Past tense	5398
Gerund or present participle	2821
Past participle	3388
3rd person singular present	3933

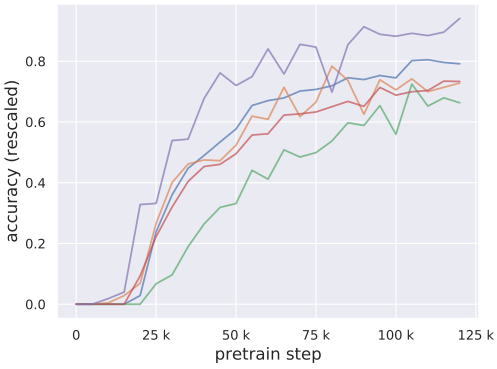
Table 4: Statistics of different verb forms used in experiments.

C.2 Mask Predict and Token Reconstruction of Different Verb Forms

We provide supplementary materials for Section 3. In Figure 7, we observe that ALBERT learns to reconstruct and predict verb of different forms at different times. The average occurrence rate of verb in different form from high to low is V-es, V-ed, V, V-en, V-ing, which coincides with the priority being leaned.



(a) Token reconstruction.



(b) Mask prediction.

Figure 7: Token reconstruction (7a) and mask prediction (7b) accuracy. We also rescale the accuracy as in Figure 1.

C.3 How Does Occurrence Frequency Affect Learning Speed of A Word?

In the main text, we observe that words of different POS are learned at different times of pretraining. We also pointed out that the learning speed of different POS roughly corresponds to their occurrence rate. However, it is not clear to what extent a word's occurrence frequency affects how soon it can be learned to reconstruct or mask-predict by the model. We provide a deeper analysis of the relationship between the learning speed of a word and its occurrence rate in Figure 8. We observe from Figure 8 that the top 50 to 99 occurring tokens are indeed learned faster than other words which occur lesser. However, as for the top 300 to 349 occurring tokens and the top 1550 to 1599 occurring tokens, it is unclear which ones are learned earlier. We can infer from Figure 8 and Figure 1b that the occurring rate and POS of a word both contribute to how soon the model can learn it to some extent.

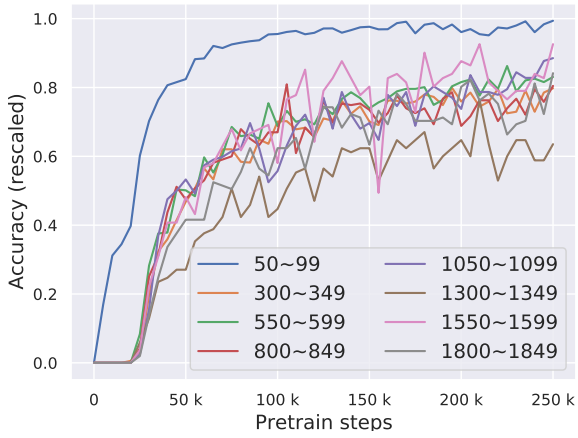


Figure 8: Rescaled mask prediction accuracy for different frequency. 50~99 means the top 50 to top 99 occurring tokens

C.4 Mask Predict and Token Reconstruction of BERT

We provide the results of BERT's token reconstruction and mask prediction in Figure 9. We observe content words are learned later than function words, while the learning speed is faster than ALBERT. To be more specific, we say a word type A is learned faster than another word type B if either the learning curve of A rises earlier than B from 0, or if the rescaled learning curve of A is steeper than that of B.

D Probing Experiments

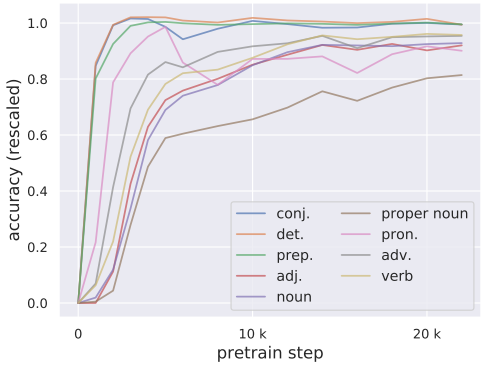
D.1 Probing Model Details

As mentioned in the main text, we modified and reimplemented the edge probing (Tenney et al., 2018) models in our experiments. The modifications are detailed as follow:

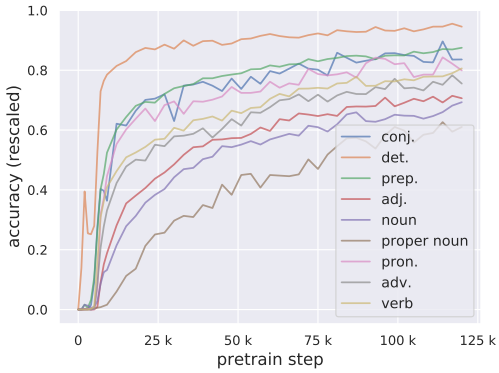
- We remove the projection layer that projects representation output from the language model to the probing model's input dimension.
- We use average pooling to obtain span representation, instead of self-attention pooling.

Task	$\lvert\mathcal{L}\rvert$	Examples	Tokens	Total Targets
POS	48	116K / 16K / 12K	2.2M / 305K / 230K	2.1M / 290K / 212K
Constituent	30	116K / 16K / 12K	2.2M / 305K / 230K	1.9M / 255K / 191K
SRL	66	253K / 35K / 24K	6.6M / 934K / 640K	599K / 83K / 56K

Table 5: Statistics of the number of labels, examples, tokens and targets (split by train/dev/test) we used in probing experiments. $\lvert\mathcal{L}\rvert$ denotes number of target labels.



(a) Token reconstruction of BERT



(b) Mask prediction of BERT

Figure 9: We also rescale the accuracy as in Figure 1b.

- We use linear classifiers instead of 2-layer MLP classifiers.
- We probe the representation of a single layer, instead of concatenating or scalar-mixing representations across all layers.

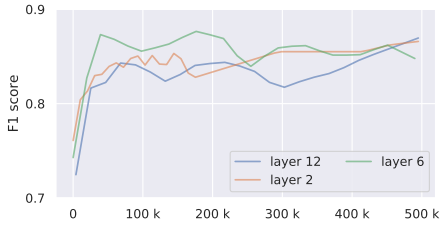
Since our probing models are much simpler than those in Tenney et al. (2018), probing results might be inferior to the original work. The number of model's parameters in our experiments is approximately 38K for POS tagging, 24K for constituent tagging, and 100K for SRL.

D.2 Dataset

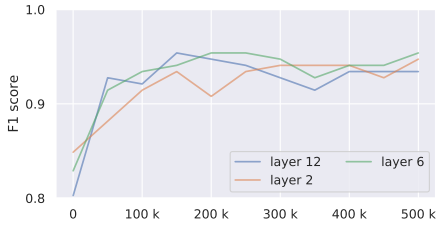
We use OntoNotes-5.0, which can be download from `https://catalog.ldc.upenn.edu/LDC2013T19`. The statistics of this dataset is in Table 5.

D.3 SRL, Coreference Resolution, and Constituent Labeling Results

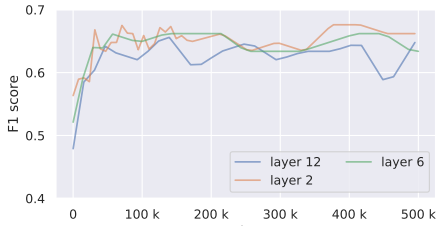
Here in Figure 10, we show supplementary figures for SRL, coreference resolution, and constituent tagging over 3 of 12 layers in ALBERT for the first 500K pretrain steps. Together with Figure 3, all four tasks show similar trends.



(a) Semantic role labeling



(b) Coreference resolution



(c) Constituent tagging

Figure 10: The probing results of SRL (10a, coreference resolution (10b) and constituency tagging (10c) during pretraining . Layers are indexed from the input layer to the output layer, so layer 2 is the output representation from layer 2 of ALBERT. Layers are indexed from 1 to 12.

Task	Examples
MRPC	3.6K / 0.4K / 1.7K
RTE	2.4K / 0.2K / 3K
STS-B	5.7K / 1.5K / 1.3K
QNLI	104K / 5.4K / 5.4K
QQP	363K / 40.4K / 391.0K
CoLA	8.5K / 1.0K / 1.1K
MNLI	392.7K / 9.8K + 9.8K / 9.8K + 9.8K
SST-2	67.4K / 0.9K / 1.8K
SQuAD2.0	13.3K / 11.9K / 8.9K

Table 6: Statistics of (train / dev/ test) in GLUE tasks and SQuAD2.0. MNLI contains matched and mismatched in dev and test set. We didn't evaluate our models' performance on test set.

D.4 Probing Results of BERT and ELECTRA

We provide the probing results of BERT and ELECTRA in Figure 11. All the probing experiments of ALBERT, BERT, and ELECTRA share the same set of hyperparameters and model architectures. We observe a similar trend as ALBERT: the probing performance rises quite quickly and plateaus (or even slightly decay) afterward. We also found that performance drop of those layers closer to ELECTRA's output layers are highly observable, which may spring from its discriminative pretraining nature.

E Downstream Evaluation

E.1 Dataset Details

We provide detail statistics of downstream tasks' dataset in Table 6. We download GLUE dataset using https://gist.github.com/W4ngatang/60c2bdb54d156a41194446737ce03e2e, and download SQuAD2.0 dataset from https://rajpurkar.github.io/SQuAD-explorer/.

E.2 Finetune Details

We use the code in https://github.com/huggingface/transformers/tree/master/examples/text-classification to run GLUE and use https://github.com/huggingface/transformers/tree/master/examples/question-answering to run SQuAD2.0. We provide detailed hyperparameters when we run GLUE benchmark and SQuAD2.0 in Table 7. We follow Liu et al. (2019b) and Lan et al. (2019), finetuning RTE, STS-B, and MRPC using an MNLI checkpoint when finetuning ALBERT. The number of parameters of all downstream tasks is close to the original ALBERT model, which is 12M.

E.3 Downstream results of ALBERT (with SQuAD2.0)

Here we provide performance of individual tasks in GLUE benchmark on development set in Figure 12, along with performance of SQuAD2.0 (Rajpurkar et al., 2018).

E.4 Downstream performance of BERT and ELECTRA

We use the same hyperparamters in Table7 to finetune BERT and ELECTRA models. Except for the performance of BERT on SQuAD2.0, all the other results are comparable with those results finetuned from the official Google pretrained models. We can observe from Figure 13 and Figure 12 that all three models' performance on downstream tasks show similar trends: Performance skyrocketed during the initial pretraining stages, and the return gradually decays later. From Figure 13c, we also find that among the three models, ALBERT plateaus the earliest, which may result from its parameter-sharing nature.

F World Knowledge Development

F.1 Dataset Statistics

In our experiment of world knowledge, we only use 1-1 relations (P1376 and P36) and N-1 relations (the rest relations in Table 8). Among those relations, we only ask our model to predict object ([Y] in the template in Table 8) that has only one token, following Petroni et al. (2019). From those relations, we report world knowledge that behaves differently during pretraining in Figure 6: we select the knowledge that can be learned during pretraining (e.g., P176), the knowledge that cannot be learned during the whole pretraining process (e.g., P140), the knowledge that was once learned and then forgotten after pretraining (e.g., P138), and knowledge that kept oscillating during pretraining (e.g., P407). The statistics of all world knowledge evaluated are in listed in Table 8.

F.2 Qualitative Results and Complete World Knowledge Results

We provide qualitative examples for Section 6 in Table 9. We also provide the complete results of all world knowledge we use in Figure 14.

	LR	BSZ	ALBERT DR	Classifier DR	TS	WS	MSL
CoLA	1.00E-05	16	0	0.1	5336	320	512
STS-B	2.00E-05	16	0	0.1	3598	214	512
SST-2	1.00E-05	32	0	0.1	20935	1256	512
MNLI	3.00E-05	128	0	0.1	10000	1000	512
QNLI	1.00E-05	32	0	0.1	33112	1986	512
QQP	5.00E-05	128	0	0.1	14000	1000	512
RTE	3.00E-05	32	0	0.1	800	200	512
MRPC	2.00E-05	32	0	0.1	800	200	512
SQuAD2.0	3.00E-05	48	0	0.1	8144	814	512

Table 7: Hyperparameters for ALBERT in downstream tasks. LR: Learning Rate. BSZ: Batch Size. DR: Dropout Rate. TS: Training Steps. WS: Warmup Steps. MSL: Maximum Sequence Length

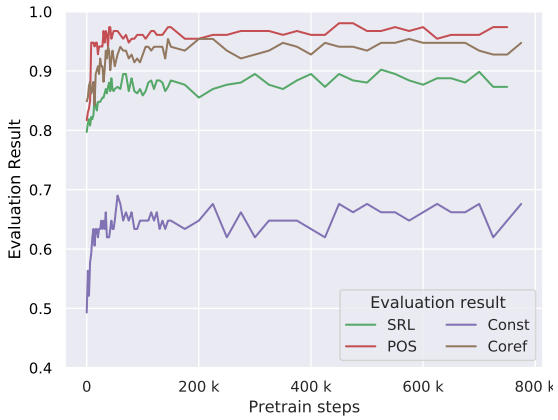
Type	Count	Template
P140	471	[X] is affiliated with the [Y] religion .
P103	975	The native language of [X] is [Y] .
P276	954	[X] is located in [Y] .
P176	946	[X] is produced by [Y] .
P264	312	[X] is represented by music label [Y] .
P30	975	[X] is located in [Y] .
P138	621	[X] is named after [Y] .
P279	958	[X] is a subclass of [Y] .
P131	880	[X] is located in [Y] .
P407	870	[X] was written in [Y] .
P36	699	The capital of [X] is [Y] .
P159	964	The headquarter of [X] is in [Y] .
P17	930	[X] is located in [Y] .
P495	909	[X] was created in [Y] .
P20	952	[X] died in [Y] .
P136	931	[X] plays [Y] music .
P740	934	[X] was founded in [Y] .
P1376	230	[X] is the capital of [Y] .
P361	861	[X] is part of [Y] .
P364	852	The original language of [X] is [Y] .
P37	952	The official language of [X] is [Y] .
P127	683	[X] is owned by [Y] .
P19	942	[X] was born in [Y] .
P413	952	[X] plays in [Y] position .
P449	874	[X] was originally aired on [Y] .

Table 8: Relations used.

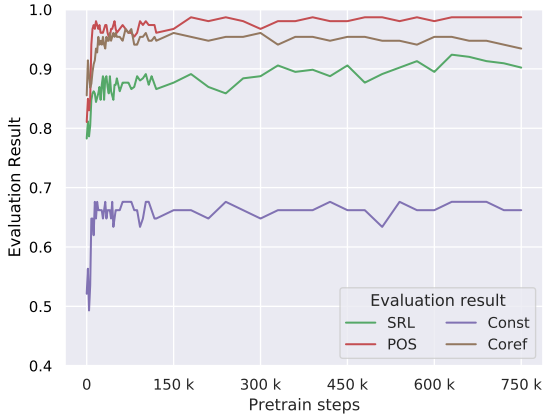
World Knowledge Prediction		
Relation	P38	P176
Query	Nokia Lumia 800 was produced by [MASK].	Hamburg airport is named after [MASK].
Answer	Nokia	Hamburg
100K	the lumia 800 is produced by nokia.	hamburg airport is named after it.
200K	nokia lu nokia 800 is produced by nokia.	hamburg airport is named after hamburg.
500K	nokia lumia 800 is produced by nokia.	hamburg airport is named after him.
1M	nokia lumia 800 is produced by nokia.	hamburg airport is named after him.

Table 9: Example results of world knowledge evolution during pretraining. We can observe that model successfully predict the object in the Nokia example since 100K steps, and doesn't forget during the rest pretraining process. On the other hand, the model is only able to correctly predict Hamburg in the second example at 200K steps, and failed to predict at other pretrain steps.

(a) Probing results of ALBERT-base model



(b) Probing results of BERT-base uncased model



(c) Probing results of ELECTRA-base model

Figure 11: Probing results of POS tagging, constituent tagging, semantic role labeling, and coreference resolution, evaluated by micro F1 score.

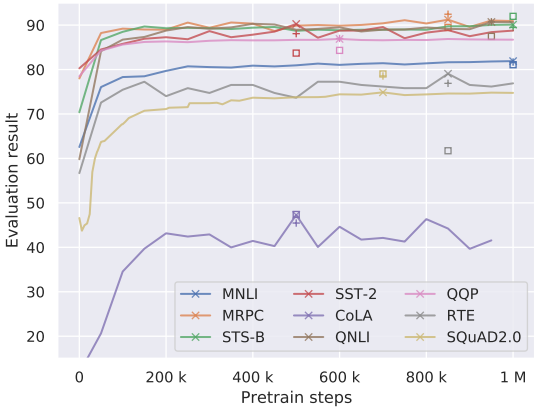
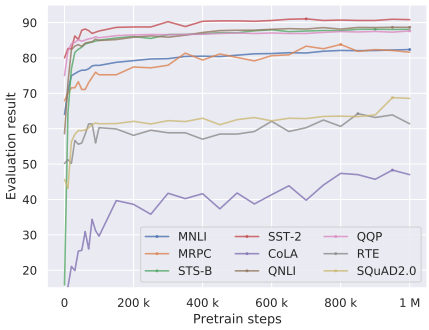
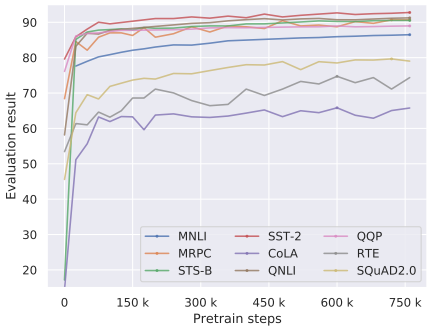


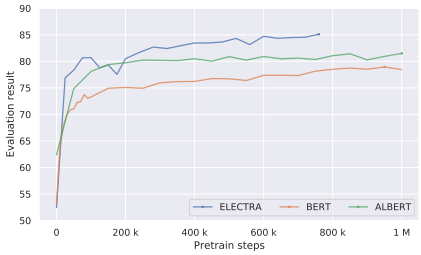
Figure 12: Performance of individual tasks in GLUE benchmark, along with SQuAD2.0 result. Best result durining pretraining marked with 'x'. Evaluation metrics: MRPC and QQP: F1, STS-B: Spearman corr., others: accuracy. The result of MNLI is the average of matched and mismatched. The result of SQuAD2.0 is the average of F1 and EM scores. Performances of albert-base-v1 and bert-base-uncased are marked with '+' and square, respectively.



(a) GLUE and SQuAD2.0 performances of BERT



(b) GLUE and SQuAD2.0 performances of ELECTRA



(c) GLUE scores of all three models

Figure 13: Performance of individual tasks in GLUE benchmark, along with SQuAD2.0 result. Best result durining pretraining marked with circle. Evaluation metrics: MRPC and QQP: F1, STS-B: Spearman corr.,others: accuracy. The result of MNLI is the averageof matched and mismatched. The result of SQuAD2.0is the average of F1 and EM scores.

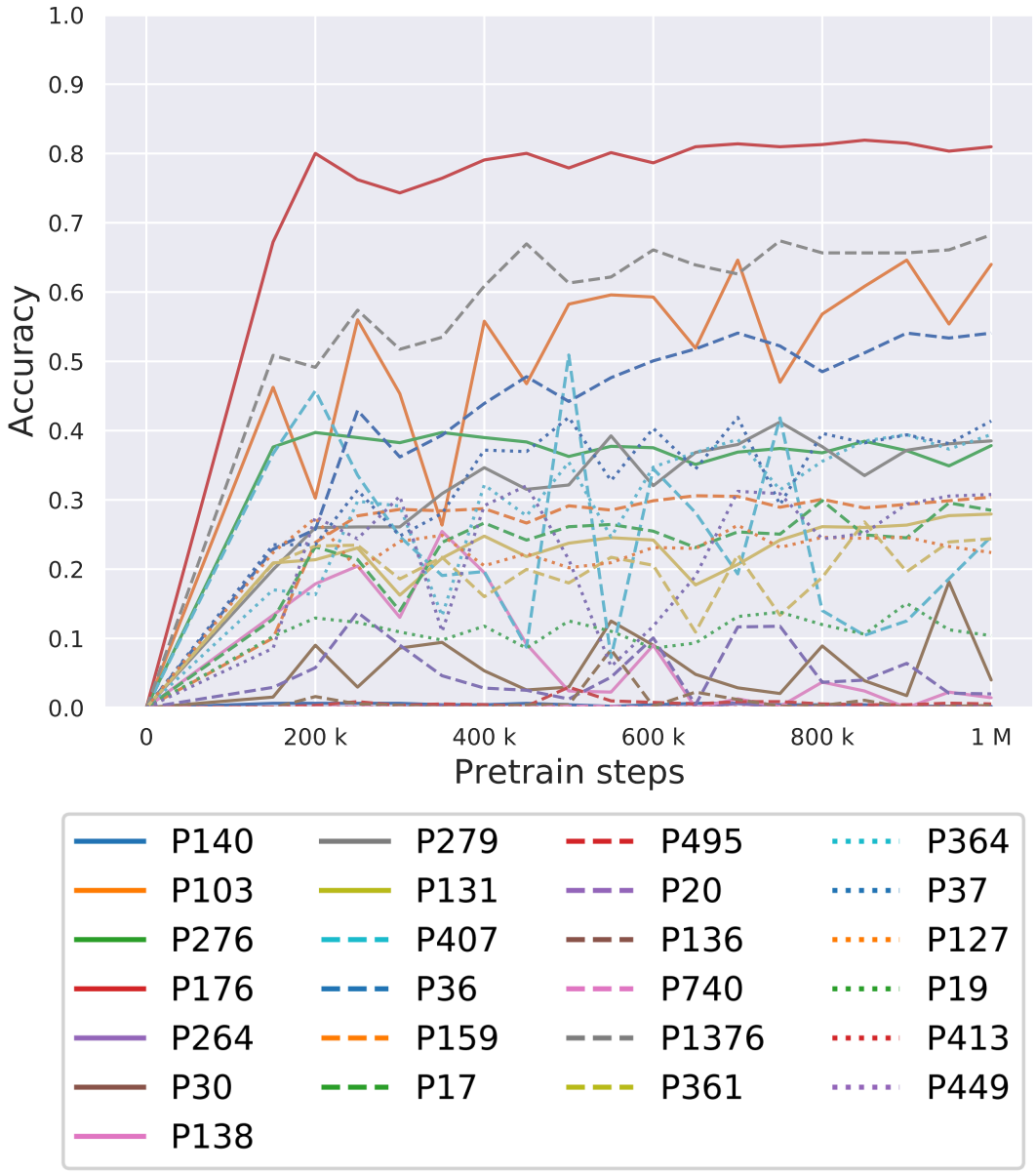


Figure 14: Prediction of all world knowledge during pretraining.