Genral response: We thank all reviewers for their constructive comments. Below is our response for common questions. **Q1. link more between the results section and your methods (R2 & R3):** Thanks for the suggestion. We will reorganize the method and experiment sections, and add more links between them in the next version.

**Q2. broader impact (R2 & R3):** For the **positive** side, as is detailed in the Broader Impact section, DynaBERT (i) alleviates concerns about the privacy by moving computation to edge; (ii) enables flexible deployment scenarios of BERT models; and (iii) is more environmentally friendly due to weight sharing. For the **negative** side, DynaBERT enables easier deployment of BERT, and thus makes the negative impacts of BERT more severe, e.g., application in dialogue systems replaces help-desks and can cause job loss. Extending our method to generative models like GPT also faces risk of generating offensive, biased or unethical outputs. We will detail these impacts in the next version.

Reviewer 1 **Q1."whether this approach can be adapted to work during the pre-training phase"**: Below we show results of using the proposed method for pre-training. Due to time limit, we only vary the width and depth of a 6-layer BERT. We compare with separately pre-trained small models in Google BERT repository (`https://github.com/google-research/bert`) and report the accuracy after fine-tuning on MNLI-m. To make sub-networks of DynaBERT the same size as those small models, for width, we also adapt the hidden state size $H = 128, 256, 512, 768$ besides attention heads and intermediate layer neurons. For depth, we adjust the number of layers to be $L = 4, 6$. As can be seen, the sub-networks of the pre-trained DynaBERT outperform separately pre-trained small networks.

| (L, H) | (6, 768) | (6, 512) | (6, 256) | (6, 128) | (4, 768) | (4, 512) | (4, 256) | (4, 128) |
|---|---|---|---|---|---|---|---|---|
| Dev set accuracy of separate small networks | 81.8 | 80.3 | 76.0 | 72.4 | 80.1 | 78.6 | 74.9 | 70.7 |
| Dev set accuracy of sub-networks of DynaBERT | 82.0 | 81.0 | 77.8 | 73.0 | 81.5 | 80.4 | 76.1 | 71.4 |

Reviewer 2 **Q1."paper quite dense and hard to read...rely on various complicated procedures", "if there is a simpler method"**: We also tried to simplify the current approach, but ablation study in Section 3.3 shows that all these various procedures helps performance and can not be removed. We will continue thinking about simplifying the method.

**Q2."Table 4, what exactly is 'fine-tuning'?"** :This is the 'fine-tuning' mentioned in Lines 138-139 in Section 2.2.

**Q3."line 140: 'In this work... I can't work out what this sentence means."**: The training in Section 2.2 consists of two parts: (i) training with augmented data and distillation objective (Lines 123-138); and (ii) fine tuning with original data and cross-entropy loss (Lines 138-141). Here 'without finetuning' means using only (i), while 'with finetuning' means using first (i) and then (ii). In the experiments in Table 1 in Section 3.1, we report the best results among all the sub-networks produced by either (i) or (ii).

**Q4. different formulation of the standard transformer layer in equation (1)**: Equation (1) shows that the attention heads can be computed in parallel and thus can be used to adjust the width of a Transformer layer.

**Q5."define TinyBERT and LayerDrop ...Why are they fair comparison points (i.e. maybe they require less fine-tuning compute power...)? "** : We will add descriptions for TinyBERT and LayerDrop. They are popular BERT compression methods, and are computationally more expensive than ours as they need to redo the pretraining step. We compare with them to show that sub-networks of DynaBERT outperform similar-sized models.

**Q6."briefly define your data augmentation procedure"**: We will add it in the final version.

**Q7."the complexity of the distillation procedure might make it harder to apply to new domains (if we need to tune many hyperparameters etc.)"**: The hyperparameters are easy and cheap to be determined. We use only a few samples to estimate the magnitude of different distillation losses, then choose $\lambda_1, \lambda_2$ to make them have similar scale.

Reviewer 3 **Q1."Table 1 is a bit overloaded and difficult to parse...which row and column are $m_w$ vs $m_d$"**: The row is depth multiplier $m_d$ taking 3 values while column is width multiplier $m_w$ taking 4 values, as defined in Line 150.

**Q2."Figure 3 is really difficult to parse too... Can you present this differently with lines corresponding to the base models?"** : We will replace the markers with lines for RoBERTa and BERT base as suggested in the final version.

**Q3."Why MNLI and SST-2 specifically for Figure 3 rather than others?"** Due to space limit, we only show plots for MNLI and SST-2 in the main content, and put those for the others in Appendix C.1 as mentioned in Line 177.

**Q4. related work**: **(1)** The primary goal of our paper is training multiple compressed sub-networks in the same model by varying width and depth. Thus we first discuss related work on compressing Transformer-based models to a certain size or various sizes by adapting only depth in Paragraph 2. Then we discuss the connection/difference between our method and others in Paragraph 3. **(2)** Thanks for providing references about the capacity of language models. In our paper, we also empirically studied capacity of DynaBERT in Section 3. From Table 1, CoLA (the task of linguistic acceptability judgments) is relatively more sensitive to the capacity. Figure 5 also shows that when reducing capacity for CoLA, the function fusion of attention heads occurs mainly in the intermediate layers. This is consistent with the finding in the recommended reference (Jawahar et al 2019) that, BERT's intermediate layers encode linguistic information. The other 3 references studied different models (e.g., character CNN and recurrent networks in Jozefowicz et al 2016; Melis et al 2017) or tasks (i.e. generation in Subramani et al 2019), and will also be discussed in the final version.

**Q5. statistical significance**: Statistical significance needs many runs of both our method and others. This is infeasible due to limited time of rebuttal. Below we report $mean \pm std$ accuracy from 5 repetitions on STS-B and SST-2, and leave more rigorous comparison for more tasks as a future work. The small std indicates the stability of DynaBERT.

| $(m_w, m_d)$ | (1x, 1x) | (1x, 0.75x) | (1x, 0.5x) | (0.75x, 1x) | (0.75x, 0.75x) | (0.75x, 0.5x) | (0.5x, 1x) | (0.5x, 0.75x) | (0.5x, 0.5x) | (0.25x, 1x) | (0.25x, 0.75x) | (0.25x, 0.5x) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| STS-B | $89.96 \pm 0.08$ | $89.39 \pm 0.08$ | $88.51 \pm 0.11$ | $89.86 \pm 0.08$ | $89.30 \pm 0.08$ | $88.46 \pm 0.09$ | $89.75 \pm 0.04$ | $89.19 \pm 0.07$ | $88.28 \pm 0.09$ | $89.16 \pm 0.06$ | $88.32 \pm 0.10$ | $87.04 \pm 0.04$ |
| SST-2 | $93.11 \pm 0.34$ | $93.16 \pm 0.16$ | $92.65 \pm 0.17$ | $93.14 \pm 0.42$ | $92.93 \pm 0.35$ | $92.51 \pm 0.26$ | $92.95 \pm 0.23$ | $92.81 \pm 0.20$ | $91.65 \pm 0.13$ | $92.64 \pm 0.34$ | $92.30 \pm 0.27$ | $91.89 \pm 0.38$ |