Training a Small-Scale BERT Model on an NLP Question Answering Task

Final Paper

Sana Moin & Nur Atiqah Zakiah Abdul Khaliq

28.01.2021

Abstract

In this paper, we aim to use a small-scale transformer-based language model, mobileBERT, to train an NLP question Answering Tasks from the bAbI dataset, provided by Facebook, and compare it with SQuAD which was used when BERT and mobileBERT were introduced. We pre-processed the dataset and then customized the model parameters in order to achieve the highest possible accuracy for the bAbI tasks 'Single Supporting Fact', 'Two Argument Relations' and 'Conjunction'. We also test BERT on these tasks, which is considered state of the art model language model for NLP, and compare our findings with mobileBERT. We conclude that mobileBERT can be adapted to solve these tasks and it gives the highest accuracy for Task 'Conjunction'.

1 Introduction

1.1 BERT

BERT [4] is a pre-trained unsupervised natural language processing (NLP) model. NLP is a field that explores how computers can be used to understand and manipulate text or speech in natural human languages in order to develop useful applications. Some of these applications include machine translation, natural language text summarization, and speech recognition [2]. The BERT model achieved state of the art performance on several NLP tasks when it was introduced. It stands for 'Bidirectional, Encoder, Representations from Transformers', transformer models such as BERT utilize multi-head attention which lets the model simultaneously attend to information from at different positions. Due to its deep bidirectional structure, BERT is able to emulate context understanding in language modeling. This is in contrast to left or right context only as well as shallow bidirectional models.

With regards to context, a technique called Masked Language Modelling (MLM) or otherwise known as a Cloze Task, which is essentially a 'fill in the blank' task, is the first task used to pre-train the BERT model. The second task is called Next Sentence Prediction (NSP), with this task, the model is given a pair of sentences and it should identify if the second sentence follows the first. From these two tasks, the BERT model is able to "understand" language and may be applied to a variety of downstream NLP tasks such as question answering (QA).

1.2 MobileBERT

Similar to BERT, MobileBERT [8] is a thin version that can be applied to various downstream NLP tasks without major changes to its architecture. MobileBERT takes advantage of bottleneck structures and a carefully designed balance between self-attentions and feed-forward networks so the necessary parameters for each layer can be significantly reduced. This results in a model that is 4.3 x smaller and 5.5 x faster than the original BERT model without sacrificing performance.

1.2.1 GLUE dataset

GLUE stands for General Language Understanding Evaluation (GLUE) benchmark [9] and is a collection of 9 NLP tasks curated to analyze the performance of NLP models on a variety of phenomena that exists in natural language. This benchmark also serves as a leaderboard for the performance of NLP models. MobileBERT is able to reach 99.2% of BERT-base's performance on GLUE, both scoring 77.7 and 78.3 respectively, with 4x fewer parameters and 5.5x faster inference on a Pixel 4 phone.

2 Question Answering in NLP

QA is well-researched within the domain of NLP and has many applications as dialog systems and entity extraction for example. QA systems are expected to automatically answer questions posed in natural language. Moreover, when given questions which no answer exists, good systems should avoid answering the question or reply that it is not possible to answer.

2.1 Question Answering Datasets

With regards to datasets for QA tasks, there are two main categories - open and closed. In such open datasets, the system is expected the generate answers based on general world knowledge in addition to provided contexts. For example, the Allen AI Science [3] and Quiz Bowl [1] datasets are both open datasets for QA tasks. Although open QA datasets emulate questions expected in real-world settings, these are applicable to more complex systems capable of significant amounts of information retrieval. On the other hand, systems are given all the necessary information to answer questions in closed datasets. The SQuAD and bAbI datasets

are two of such closed datasets and are the two datasets we will pay close attention to in this seminar paper.

2.2 SQuAD Dataset

SQuAD stands for 'Stanford Question Answering Dataset' and it contains 100,000+ questions posed by crowdworkers on a set of Wikipedia articles. The answers to each question are a segment from a corresponding 'context' which is a passage of text.

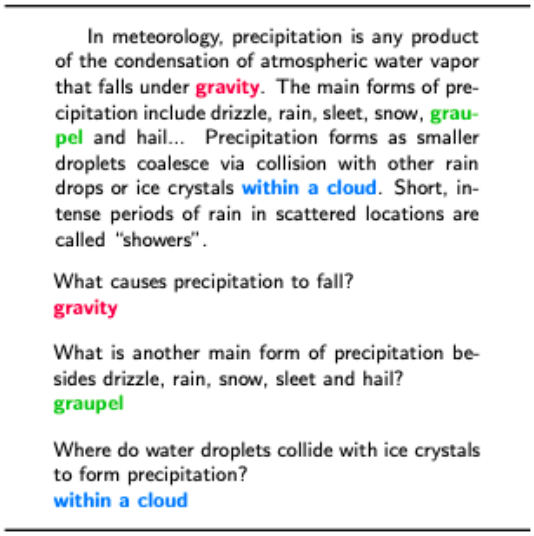

Figure 1: An example from the SQuAD dataset [7]

It essentially tests the system's ability to read a passage of text and then answer questions about it. SQuAD is a popular dataset for researchers to test their NLP systems and there are several reasons for this. It is a big dataset that makes it suitable for complex models that require large amounts of training data. Additionally, it tests several types of reasoning including lexical variation, where the system should recognize synonyms and have enough general knowledge, and multi-sentence reasoning, where the system needs to derive the answer from more than one sentence in the given context.

In our reference paper [8], researchers had trained their system mobileBERT on the SQuAD dataset version 1.1. This version does not contain unanswerable questions. The examples from a SQuAD dataset was formatted into .json files to be given as input to the system for training and validation. With regards to the structure of the information within the datasets, the examples can be broken down into blocks. Each block can be identified by its title and it consists of several context passages. Each context passage has 5 corresponding questions and answer sets. Each question and answer set has an id. The question is represented by a text string and answers are represented by both a text string as well as the index of the first character of the answer in the context. See Listing 1 in the Appendix to see how the information was structured for SQuAD version 1.1.

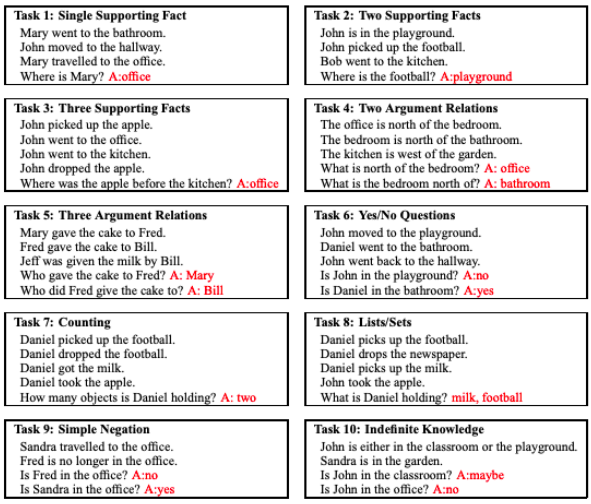

Figure 2: Examples of tasks from the bAbI dataset [10]

2.3 BAbI Dataset

To address the AI question answering task, Facebook has provided a set of 20 tasks in their bAbI dataset. The tasks measure comprehension in multiple ways - chaining facts, simple induction, and deduction for example [10]. Tasks selected were meant to be natural and "easy" to a human, no knowledge in formal semantics, machine learning, logic, or knowledge representation should be necessary for a human adult fluent in the language. With that, the data was stated to be noiseless and a human adult should have no problem attaining 100% accuracy on the questions. The data itself was produced using a simple simulation of characters and objects moving around and interacting in locations. The tasks were set up so that the right answers are either one-word answers or a short list of words to make evaluation straightforward. In the data release, in addition to providing the above 20 tasks in English, they also provide them (i) in Hindi; and (ii) with shuffled English words so they are no longer readable by humans.

Researchers wanted to keep the set-up similar to software testing in computer science. Each task is a "leaf" test case and is independent of other test cases as much as possible. The structure of the tasks in bAbI dataset is different than that of the SQuAD dataset. In the bAbI dataset, each block consists of 15-20 associated sentences, depending on the task, and the sentences alternate between statements and questions. Every third sentence is a question and the context, from which the answer comes, differs for each question as it is every statement that comes before that question. For example, for the question in sentence 3, the context is sentence 1 and 2 while the context for the question in sentence '6' is sentence 1,2,3,4 and 5. See Listing 3 in the Appendix for an example from the "Single Supporting Fact" Task.

3 Problem Statement

The aim of this seminar project was to train a small-scale BERT model on an NLP task. BERT in itself is one of the most path-breaking development in the field of NLP but due to its extensive computation requirements, it is difficult to make use of it for extending it. MobileBERT was our model of choice because relative to BERT it is small and fast without compromising on the performance. MobileBERT comes with pre-trained models on datasets such as SQuAD and we would test our model on several tasks from the bAbi dataset. Facebook's bAbI dataset was chosen because it is still a closed dataset for QA but the structure of the QA tasks are different than that of SQuAD which the model was already trained on. This would allow us to have more extensive conclusions about mobileBERT's reasoning capabilities in natural language. Therefore, we will work with mobileBERT. We define our problem statement as follows:

Is it possible to adapt mobileBERT to the bAbi dataset?

3.1 Scope

For the scope of this paper, the QA tasks that mobileBERT was trained on was limited due to the time constraints on the seminar project. Our aim was to work incrementally, starting with what seemed like the easiest tasks and then working our way to more difficult tasks. In this seminar project, we focused on tasks from the bAbI dataset where questions could be answered with a single word answer and derived from one sentence within the corresponding context. This would also make the evaluation of the models much simpler as we could directly check if the one-word answer was correct by matching it to the target output. Based on that, we would be training on tasks 1, 4 and 12 from the bAbI dataset which are 'Single Supporting Fact, 'Two Argument Relations' and 'Conjunction' tasks respectively.

3.2 Expected Performance

The expectations of mobileBERT's performance on the bAbI tasks were primarily based on the knowledge required for the task. For Task 1, the 'Single Supporting Fact' task (see Listing 3 in the Appendix), the model is not required to have any world knowledge as all the information required to answer the question can be found from the context. However, the questions are asked throughout the context and 'when' the question is asked may change the answer to the question. The model has to update the location of a character when it sees a new sentence. Compared to the questions from the SQuAD dataset, this task would be more difficult because it is more than word matching or entity recognition although researchers claim this could be one of the simplest tasks [10]. As such, it would be expected that mobileBERT performs worse on this task than on the SQuAd dataset.

For Task 4, the 'Two Argument Relations' Task (see Listing 3 in the Appendix),

it would be required that the model pieces together the information from the two supporting sentences. Additionally, some world knowledge is also expected to paint a bigger picture of the story from the sentences as the model would need to understand what the four cardinal directions are and how they relate to each other. MobileBERT would be expected to fare worse on this task than on the 'Single Supporting Fact' task.

Task 12, 'Conjuntion' task, is a simple extension of the 'Single Supporting Fact' task where sentences contain information about two characters instead of just one. We would expect the model to perform similar to the 'Single Supporting Fact' task because the tasks are very similar.

4 Method

4.1 Environment and Set-up

We set up several python 3 (up to 3.7) Google Colab Notebooks for running mobileBERT and BERT models so that the computing resources offered by Google Colab could be accessed and the models could be tested simultaneously. The supported Tensorflow version that should be used is 1.15. The mobileBERT source code consists of the following python files:

- modeling.py - contains bert config and model
- optimization.py - contains Layer-wise Adaptive Moments optimizer for Batch training and AdamWeightDecayOptimizer
- run_classifier.py - bert fine tuning runner, contains classes for various dataset processors, input example and features and methods for working with datasets
- tokenization.py - contains FullTokenizer, BasicTokenizer and WordpieceTokenizer classes
- run_squad.py - used for running Squad dataset on mobileBERT model.
- Other supporting python files

4.1.1 Example: Set-Up on bAbI

- Dataset used for training and prediction were downloaded from `https://research.fb.com/downloads/babi/`
- Number of questions in each task from bAbI dataset for training - 1000
- Number of questions in each task from bAbI dataset for testing - 1000
- Command for running mobilebert on this dataset- python3 run_babi.py –list_of_parameters

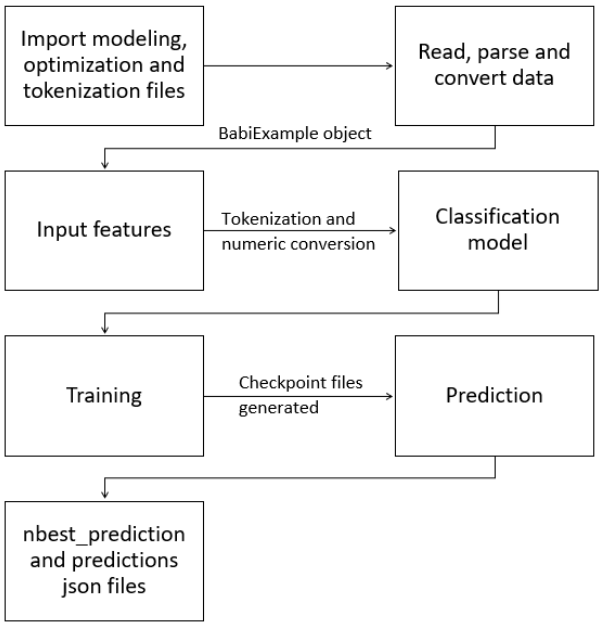

Figure 3: Execution flow of mobileBERT for babi dataset

- List of parameters are shown in Table 2 in the Appendix

4.1.2 Execution flow using run_babi.py

This execution flow is the same as run_squad that comes with mobileBERT source code. The modeling, optimization and tokenization files are first imported. The training data, that is in JSON form, is read, parsed and created into an object of BabiExample which stores information for further processing. These example object consists of qas_id, question_text, doc_tokens, answers, orig_answer_text, start_position, end_position and is_possible. These example objects are then processed to convert them into input features. For that, the various text fields are tokenized and converted to the appropriate form and eventually returned as numerical feature data for processing into the classification model. Next, the classification model is created and training begins. The checkpoint files are generated after training and stored. Then the predictions are performed on the prediction dataset. After prediction, two JSON files are generated, nbest_predictions and predictions.

The nbest_predictions file consists of a set of predictions for every answer with their text, probability, start_logit and end_logit. The predictions file consists of the best answer for every question given. The same flow is also shown in figure 3

4.2 Pre-Processing of bAbI Dataset

Before we could give the tasks from a bAbI dataset as input to mobileBERT, there was some pre-processing that had to be done. We decide to make the structure like that of Squad dataset which we can see in Listing 1. As shown in the Listing 2 and described in section 2.3 'BAbI dataset', the sentences are structured in a block, which we can consider as the context of the questions that come across in the block. The answer to the questions is a single word and have an associated line number with them, which point to the offset of the answer.

To make the dataset meaningful to mobileBERT, we write a python code to convert this data into a JSON form. This JSON would contain the dataset values as different paragraphs. Each para(single dataset) in the paragraphs would have a context, a qas section which contains the answer offset and answer text, a question and an id of the para. We consider the context as the set of sentences that occur before the question in the particular block. The answer is written as a single word in the sentence after the question, followed by the offset of the sentence this answer belongs to. Using the sentence offset of the answer and the answer text, the answer offset is calculated from the context. All these details are taken and stored as a single para and the set of these paras creates our paragraph object in the JSON, which we take as our training and test data. The python script(convert_babi.py) we wrote is responsible for converting the existing bAbI data text to the JSON structure as described. The final JSON structure can be seen in Listing 3.

Additionally, the vocab.txt file from the mobileBERT source code was extended with unique words that exist in the bAbI training as well as testing files but was absent in the vocab.txt file prior.

4.3 Experimental Set-up

Given the objective to train mobileBERT on QA tasks from the bAbI dataset, our goal with regards to the experiments was to identify the most optimal hyperparameters for mobileBERT on each of the three tasks. For each task, we manipulated three hyperparameters namely the number of epochs, the learning rate and the training batch size as the researchers had recommended [8]. In addition to mobileBERT, we also trained BERT on the same QA tasks from the bAbI dataset. This was done in order to verify the claims researchers had made about the performance of mobileBERT which should be comparable to BERT and to obtain a baseline measure for mobileBERT's performance on QA tasks. The experiments were set up in a way to be exploratory in nature as it was not possible to use an algorithm such as grid search to find the most optimal hyperparameters. The hyperparameter optimization is done manually and that allowed us more freedom when manipulating the hyperparameter values, and the process would be driven by the performance of the model. For example, if we would found that decreasing the learning rate would improve the performance when we would explore learning

rates with small values. Additionally, it should be noted that we did the experiments on bare versions of mobileBERT and BERT as we did not determine how to use any pre-trained versions of mobileBERT within the time constraints of this seminar project and wanted to keep it consistent across both models.

Additionally, the accuracy of the models was used to evaluate their performance on the tasks. We felt that accuracy would be the most useful metric for QA tasks as we were interested in how many predicted answers matched the target answers. Additionally, this was what other researchers who had worked on the bAbI dataset had reported [5, 6, 10]

5 Results and Discussion

Performance on the bAbI dataset
We found that mobileBERT could be trained to solve the aforementioned tasks from the bAbI dataset. The mobileBERT model was able to achieve 52.3% accuracy with 5 epochs, a learning rate of 4E-05 and a training batch size of 8 on the Single Supporting Fact task. With these same parameters, BERT achieved 52.6% accuracy. For the 'Two Argument Relation' task, mobileBERT achieved a higher accuracy of 55.8% with 10 epochs, a learning rate of 4E-05 and a training batch size of 8 and BERT achieved a comparable result of 56.3% accuracy. For task 12, it was found that the learning rate had significantly boosted the performance of mobileBERT as it achieved an accuracy of 75.4% with 20 epochs, 4E-06 and a training batch size of 16. With the same number of epochs, BERT achieved an accuracy of 69.8%. We did not experiment with the learning rate of BERT since we only wanted to use it as a baseline measure and changing the training batch size to 16 gave us errors.(Conventions used on the figure titles- MB: mobileBERT, B: BERT, lr: Learning rate)

Comparing the results to our expectation, we found that our results did mostly match our expectations about the difference between the datasets. MobileBERT did perform worse on bAbI than on SQuAD as logical reasoning is more difficult than word matching. However, we noticed some peculiarities when we compared the results across each task. MobileBERT performed worse on the 'Single Supporting Task' task where we expected it to perform the best among the three tasks. Additionally, mobileBERT performed better on the 'Conjunction' task than on the 'Single Supporting Fact' task, and even if we control 'Single Supporting Fact' using the same hyperparameters, the accuracy does not improve for it. Moreover, mobileBERT was still able to do fairly well on the 'Two Argument Relation' task even though logical reasoning, as well as world knowledge, was required.

Effects of Hyperparameters It was observed that the number of epochs required for the model to start getting good results was different for each task and in turn, the optimal number of epochs also differed across the tasks (Table 1).

Table 1: Results for mobileBERT and BERT

Task	Model	Epoch	Learning Rate	Training Batch Size	Accuracy	Loss
1	mobileBERT	5	4E-05	8	52.3	1.23
	BERT	5	4E-05	8	52.6	1.45
4	mobileBERT	10	4E-05	8	55.8	0.72
	BERT	10	4E-05	8	56.3	0.70
12	mobileBERT	20	4E-06	16	75.4	0.59
	BERT	20	4E-05	8	69.8	0.55

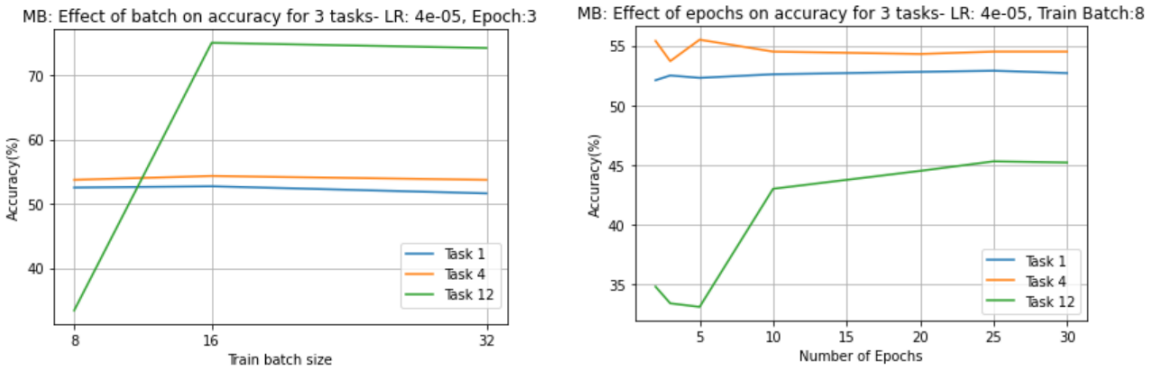

Figure 4: **Left**: Train batch size vs accuracy for mobileBERT (lr=4e-05, epoch=3). **Right**: Accuracy vs Epoch for mobileBERT (lr=4e-05, batch_size=8).

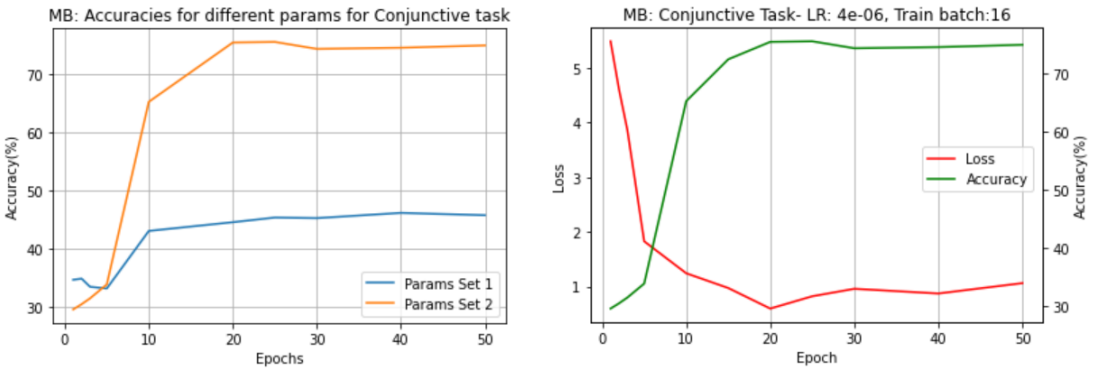

Figure 5: **Left**: Accuracy Vs Epoch for mobileBERT on Task 12 using 2 different learning rates (Params Set 1: lr=4E-05, Params Set 2: lr=4E-06). **Right**:Accuracy/Loss Vs Epoch on for mobileBERT on Task 12 using the smaller learning rate.

The performance for tasks 1 and 4 reached its peak early on, for epoch 5 and 10 respectively, and trying larger epoch values did not boost mobileBERT's performance on them(Fig 4 Right). On the other hand, mobileBERT performed poorly on task 12 with smaller epoch values but the performance significantly improved

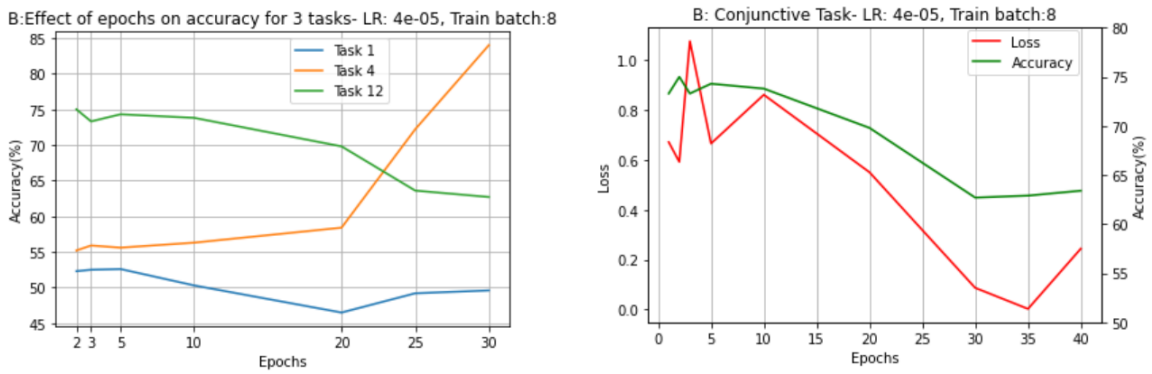

Figure 6: **Left**: Accuracy vs Epoch for BERT. **Right**: Accuracy/Loss vs Epoch on the Task 12 for BERT(lr=4e-05, batch_size=8)

with more than 10 epochs and achieved the best results at 20 epochs. We see a similar pattern with the training batch sizes. The training batch sizes made no significant improvements to the performance of mobileBERT in tasks 1 and 4 but for task 12, training sizes of 16 and 32 significantly improved its accuracy(Fig 4 Left). However, we see that learning rates influence the performance of mobileBERT for both tasks 4 and 12. Smaller learning rates boosted the model's performance and this was especially true for task 12, where the performance almost doubled(Fig 5 Left). We can also see how loss and accuracy varied for Task 12 across the epochs when using the most optimal hyperparameters to test(Fig 5 Right).
For BERT, we could see that the performance boosted for Task 4 after 20 epochs, reduced for Task 12 with an increase in epoch and had a negative effect for Task 1 for epochs between 5 to 25(Fig 6 Left). The loss and accuracy for Task 12 are shown in Fig 6 Right, where no significant pattern of accuracy and loss were observed. (refer Table 3, 4 and 5 in Appendix for complete test results).

6 Conclusion and Future Work

In conclusion, we found mobileBERT could be adapted to solve tasks from the bAbI dataset. Even though it had poorer performance on the bAbI dataset, this does not prove that mobileBERT is inferior because the questions from the tasks from the bAbI dataset do require logical reasoning and cannot merely be solved by word matching with the given context unlike questions from the SQuAD dataset. Additionally, it could also be noted that mobileBERT is capable of 'understanding' a language. This is because the predicted answers from mobileBERT always 'made sense' even though they may not be correct. The predicted answers were actual single word answers and were names of places as the question was always asked for a location. From the performance of mobileBERT on the bAbI dataset, it would also be safe to say that mobileBERT is a thinner version of BERT indeed with comparable performances as researchers had claimed [8]. We can also conclude that bAbI in itself has more difficult tasks than SQUAD as we could not achieve a

performance of more than 85%, which was observed for different versions of mobileBERT, using both BERT and mobileBERT on any task among all tests performed.

Our research has demonstrated the capabilities of mobileBERT and so with regards to future work, we suggest looking into using pre-trained models of mobileBERT and perform transfer learning to further improve the performance of the model. Additionally, future work could include training mobileBERT onto all tasks from the bAbI dataset to have a more comprehensive understanding of its performance on bAbI as a whole.

References

[1] Jordan Boyd-Graber et al. "Besting the quiz master: Crowdsourcing incremental classification games". In: *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*. 2012, pp. 1290–1301.

[2] Gobinda G Chowdhury. "Natural language processing". In: *Annual review of information science and technology* 37.1 (2003), pp. 51–89.

[3] Peter Clark and Oren Etzioni. "My computer is an honor student—But how intelligent is it? Standardized tests as a measure of AI". In: *AI Magazine* 37.1 (2016), pp. 5–12.

[4] Jacob Devlin et al. "Bert: Pre-training of deep bidirectional transformers for language understanding". In: *arXiv preprint arXiv:1810.04805* (2018).

[5] Divyansh Kaushik and Zachary C Lipton. "How much reading does reading comprehension require? a critical investigation of popular benchmarks". In: *arXiv preprint arXiv:1808.04926* (2018).

[6] Ankit Kumar et al. "Ask me anything: Dynamic memory networks for natural language processing". In: *International conference on machine learning*. PMLR. 2016, pp. 1378–1387.

[7] Pranav Rajpurkar et al. "Squad: 100,000+ questions for machine comprehension of text". In: *arXiv preprint arXiv:1606.05250* (2016).

[8] Zhiqing Sun et al. "Mobilebert: a compact task-agnostic bert for resource-limited devices". In: *arXiv preprint arXiv:2004.02984* (2020).

[9] Alex Wang et al. "Glue: A multi-task benchmark and analysis platform for natural language understanding". In: *arXiv preprint arXiv:1804.07461* (2018).

[10] Jason Weston et al. "Towards ai-complete question answering: A set of prerequisite toy tasks". In: *arXiv preprint arXiv:1502.05698* (2015).

A Appendix

Listing 1 JSON example for SQuAD Dataset

```
{"data": [
      {"title": "University_of_Notre_Dame",
      "paragraphs": [
         {"context": "Architecturally, ...",
          "qas": [
                {"answers": [{
                  "answer_start": 515,
                  "text": "Saint Bernadette Soubirous"}],
                 "question": "To whom did ...",
                 "id": "5733be284776f41900661182"},
                {"answers": [{
                  "answer_start": 188,
                  "text": "a copper statue of Christ"}],
                 "question": "What is in ...?",
                 "id": "5733be284776f4190066117f"},
                {"answers": [{
                  "answer_start": 279,
                  "text": "the Main Building"}],
                 "question": "The Basilica ...",
                 "id": "5733be284776f41900661180"},
                {"answers": [{
                  "answer_start": 381,
                  "text": "a Marian place ..."}],
                 "question": "What is ...",
                 "id": "5733be284776f41900661181"},
                {"answers": [{
                  "answer_start": 92,
                  "text": "a golden statue ..."}],
                 "question": "What sits ...?",
                 "id": "5733be284776f4190066117e"}
                ]},
                ...,
  "version": "1.1"}
```

Listing 2 JSON example for BAbI Dataset

```
{
  "data": [
    {
      "title": "Babi Dataset qa6",
      "paragraphs": [
        {
          "context": "John travelled to the hallway.
          Mary journeyed to the bathroom.",
          "qas": [
            {
              "answers": [
                {
                  "answer_start": 22,
                  "text": "hallway"
                }
              ],
              "question": "Where is John?",
              "id": "0_0"
            }
          ]
        },
        {
          "context": "John travelled to the hallway.
          Mary journeyed to the bathroom.
          Daniel went back to the bathroom.
          John moved to the bedroom.",
          "qas": [
            {
              "answers": [
                {
                  "answer_start": 53,
                  "text": "bathroom"
                }
              ],
              "question": "Where is Mary?",
              "id": "0_1"
            }
          ]
        }
      ]
    }
```

Table 2: Default Parameters used to run squad dataset on mobileBERT

Parameters	Values
bert_config_file	config/uncased_L-24_H-128_B-512_A-4_F-4_OPT.json
data_dir	${DATA_DIR}
doc_stride	128
init_checkpoint	${INIT_CHECKPOINT}/mobilebert.ckpt
learning_rate	4e-05
max_answer_length	1
max_query_length	64
max_seq_length	384
n_best_size	20
num_train_epochs	5
output_dir	${OUTPUT_DIR}
predict_file	babi_data$_{c}$*onverted_t*1*_train.json*
train_batch_size	32
train_file	babi_data$_{c}$*onverted_t*1*_test.json*
vocab_file	${INIT_CHECKPOINT}/vocab.txt
warmup_proportion	0.1

Listing 3 Snippet of the Single Supporting Fact Task from the bAbI Dataset

```
1 John travelled to the hallway.
2 Mary journeyed to the bathroom.
3 Where is John?     hallway    1
4 Daniel went back to the bathroom.
5 John moved to the bedroom.
6 Where is Mary?     bathroom    2
7 John went to the hallway.
8 Sandra journeyed to the kitchen.
9 Where is Sandra?     kitchen    8
10 Sandra travelled to the hallway.
11 John went to the garden.
12 Where is Sandra?     hallway    10
13 Sandra went back to the bathroom.
14 Sandra moved to the kitchen.
15 Where is Sandra?     kitchen    14
```

Listing 4 Snippet of the Two Argument Relations Task from the bAbI Dataset

```
1 The hallway is east of the bathroom.
2 The bedroom is west of the bathroom.
3 What is the bathroom east of?    bedroom    2
1 The bedroom is west of the kitchen.
2 The hallway is west of the bedroom.
3 What is west of the kitchen?    bedroom    1
1 The bathroom is north of the garden.
2 The hallway is north of the bathroom.
3 What is north of the garden?    bathroom    1
```

Listing 5 Snippet of the Conjunction Task from the bAbI Dataset

```
1 John and Mary travelled to the hallway.
2 Sandra and Mary journeyed to the bedroom.
3 Where is Mary?     bedroom    2
4 Mary and Daniel travelled to the bathroom.
5 Daniel and Sandra journeyed to the office.
6 Where is Mary?     bathroom    4
7 Daniel and Mary went to the bedroom.
8 Daniel and Sandra travelled to the hallway.
9 Where is Sandra?     hallway    8
10 Mary and Sandra journeyed to the garden.
11 Sandra and Mary travelled to the hallway.
12 Where is Mary?     hallway    11
13 Daniel and John journeyed to the bathroom.
14 Daniel and Mary went back to the office.
15 Where is John?     bathroom    13
```

Table 3: Test Results for Task 1

Test	Num Epochs	Learning Rate	Test Batch Size	Accuracy- MobileBert	Accuracy- Bert
test 1	2	4.00E-05	8	52.1	52.3
test 2	3	4.00E-05	8	52.5	52.5
test 3	3	2.00E-05	8	35.6	-
test 4	3	4.00E-06	8	51.4	51.7
test 5	3	4.00E-07	8	33.6	43.2
test 6	3	4.00E-04	8	52.2	3
test 7	3	4.00E-05	16	52.7	-
test 8	3	4.00E-05	32	51.6	-
test 9	3	4.00E-06	16	35.6	-
test 10	3	2.00E-05	8	40.7	-
test 11	3	9.00E-05	8	52.3	-
test 12	5	4.00E-05	8	52.3	52.6
test 13	5	7.00E-05	8	52.3	-
test 14	5	9.00E-05	8	52.8	-
test 15	10	4.00E-05	8	52.7	50.3
test 16	20	4.00E-05	8	52.8	46.5
test 17	20	4.00E-06	16	42	-
test 18	25	4.00E-05	8	52.9	49.2
test 19	25	4.00E-07	8	30.3	-
test 20	30	4.00E-05	8	52.7	49.6

Table 4: Test Results for Task 4

Test	Num Epochs	Learning Rate	Train Batch Size	Accuracy- MobileBert	Accuracy- Bert
test 1	2	4.00E-05	8	55.4	55.2
test 2	3	4.00E-05	8	53.7	55.9
test 3	3	4.00E-06	8	54.2	55.8
test 4	3	2.00E-07	8	55.9	-
test 5	3	4.00E-07	8	55.9	55.8
test 6	3	4.00E-04	8	54	3.1
test 7	3	4.00E-05	16	54.3	-
test 8	3	4.00E-05	32	53.7	-
test 9	5	4.00E-05	8	55.5	55.6
test 10	10	4.00E-05	8	55.8	56.3
test 11	20	4.00E-05	8	54.3	58.4
test 12	25	4.00E-05	8	54.5	72.2
test 13	30	4.00E-05	8	54.5	84

Table 5: Test Results for Task 12

Test	Num Epochs	Learning Rate	Train Batch Size	Accuracy- MobileBert	Accuracy- Bert
test 1	1	4.00E-06	16	29.5	-
test 2	2	4.00E-05	8	34.8	75
test 3	2	4.00E-06	16	30.4	-
test 4	3	4.00E-05	8	33.4	73.3
test 5	3	4.00E-06	8	70.2	73.7
test 6	3	4.00E-04	8	74.2	0.7
test 7	3	4.00E-05	16	75	-
test 8	3	4.00E-05	12	74.6	75
test 9	3	4.00E-05	32	74.2	-
test 10	3	4.00E-06	16	31.4	-
test 11	5	4.00E-05	8	33.1	74.3
test 12	5	4.00E-06	8	73.5	73.3
test 13	5	4.00E-06	16	33.8	-
test 14	10	4.00E-05	8	43	73.8
test 15	10	4.00E-06	8	74.3	73.7
test 16	10	4.00E-06	16	65.2	-
test 17	15	4.00E-06	16	72.4	-
test 18	20	4.00E-05	8	44.5	69.8
test 19	20	4.00E-06	16	75.4	-
test 20	25	4.00E-05	8	45.3	-
test 21	25	4.00E-06	16	75.5	-
test 22	25	4.00E-05	8	63.6	
test 23	30	4.00E-05	8	45.2	-
test 24	30	4.00E-06	16	74.3	-
test 25	30	4.00E-05	8	62.7	
test 26	35	4.00E-05	8	62.9	
test 27	40	4.00E-05	8	46.1	-
test 28	40	4.00E-06	16	74.5	-
test 29	40	4.00E-05	8	63.4	
test 30	50	4.00E-05	8	45.7	-
test 31	50	4.00E-06	16	74.9	-