Mini Bert Optimized for Multi Tasks

Stanford CS224N Default Project

LIN LIN
Department of Computer Science
Stanford University
linwoods@stanford.edu

Abstract

In this study, we explored the intricacies of fine-tuning BERT, a state-of-the-art transformer model, for multitasking across various NLP domains such as sentiment analysis, paraphrase detection, and semantic textual similarity. Our findings illuminate the inherent challenges associated with multitask learning, including task interference, data imbalance, and the complexity of model architecture and loss function management.

1 Acknowledgement

- Mentor: Arvind Venkat Mahankali

2 Introduction

BERT(Bidirectional Encoder Representations from Transformers) is an Encoder based neural network introduced by researchers at Google (Devlin et al., 2019). It has demonstrated transformative impact in the field of Natural Language Processing(NLP). Different from traditional Word2Vec or GloVe static representations, BERT utilized dynamic, contextualized embeddings for each word, interpreted as token. The embeddings allow carrying richer understanding of complex language syntax and capturing nuances, which makes it a better option in handling basic NLP tasks like Sentimental Analysis, Sentence similarity comparison.etc. Multi task language models which are designed with the ability to handle multiple NLP tasks simultaneously comes with many challenges, including conflict of gradients, overfitting, data imbalance.etc. We will discuss how it's handled in other researches and our implementation over it.

3 Related Work

Multi task language models which are designed with the ability to handle multiple NLP tasks simultaneously comes with many challenges. They have been a wide range of research works proposed in this area.

Multi-Task Learning problem started with the intention to develop transformable knowledge for one model to handle different tasks. Ideally, we also hope training on different tasks would benefit each other. Large Language Model has demonstrated great universal language representation by training on large amount of unlabeled data. Elmo proposed by (Peters et al., 2018) and GPT proposed by (Radford and Narasimhan, 2018).etc have all demonstrated great capability in handling generic language tasks in English context.

Multiple solutions have been presented by modifying BERT model with extra task specific layers and parameters to handle the multi tasks. Adapter (Houlsby et al., 2019) is introduced a concept of small learnt bottleneck layers inserted within each layer of a pre-trained model. Each adapter is trained for one downstream task and (Pfeiffer et al., 2020) also introduced a universal interface enabling shoring and sharing the adapters.

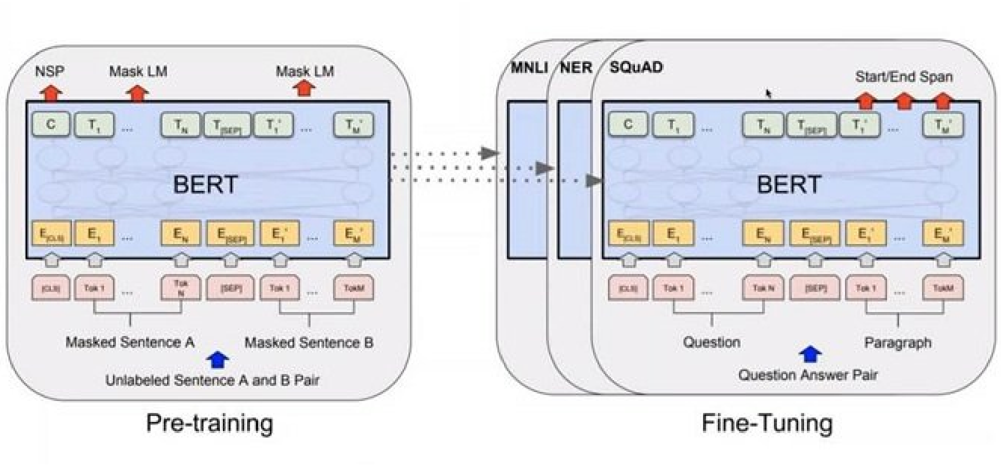

Figure 1: BERT Pre-Training and Finetuning for individual task

To share the majority of the weights of BERT model but still be able to optimized efficiently towards each task, (Stickland and Murray, 2019) has presented works of Projected Attention Layers. By inserting task-specified layers with parameters into BERT, PAL architecture is able to achieve state of art performance across multiple tasks.

Multiple tasks work from(Liu et al., 2019) is also promising. They presented a multi-Task Deep Neural Network(MT-DNN) framework leverages large amounts of cross-task data. By extending on (**?**) with a pre-trained BERT model and they demonstrated good result on SNLI, SciTail and GLUE tasks. To handle the overfitting problem comes with the long finetuning, SMART Regularization (Jiang et al., 2020) has been introduced. They are good inspirations for our project. (Bi et al., 2022) work is an extension on similar idea.

4 Approach

4.1 Design decisions

While finetuning a model for multitasks, sometime we will observe performance degradation compared with single tasks. There are multiple factors behind it.

- **Task Interference**: The finetuning process on multiple tasks requires to find a balance between different objectives. If tasks have conflicting requirements, both of the task performance would be sub-optimal. Multiple researches have been done on it, (Yu et al., 2020) describes the conflict of gradient across multiple tasks and how to adjust them to increase efficiency.
- **Data Imbalance**: Multi-task fine-tuning involves combining datasets from different tasks. If one task has significantly more data than others, the model would be trained towards a single task heavily. We went with oversampling on the train dataset when training set accuracy is still low for the relative smaller dataset to compensate.
- **Hyperparameter Tuning**: Optimal hyperparameters maybe different for different task head. We introduced multiple options of learning rate scheduler to control the step size taken in the direction of gradient descent. Also a early-stop strategy is implement to stop training for the tasks that are not showing improvements.
- **Regularization**: Extensive finetuning job is prune to overfitting problem, which could be observed when achieving high accuracy score in Training dataset, having much lower score on DEV and TEST dataset. The SMART Regularization deployed a additional loss factor $R_{\mathrm{s}}(\theta)$,

$$min_{\theta} F(\theta) = L(\theta) + \lambda_{\mathrm{s}} R_{\mathrm{s}}(\theta)$$

where

$$L(\theta) = \frac{1}{n}\sum_{i=1}^{n} \ell(f(x_i;\theta), y_i)$$

We customized a SMART Regularization under(Jiang et al., 2020) in this project. However, there are multiple ways to integrate with SBERT pooling and we may need a more resources in covering this part.

4.2 BERT Model

First, A baseline model is built based on BERT-base-uncased model from HuggingFace. The model is pretrained and finetuned on classify sentence sentiment(SST-5) data and demonstrated great performance for DEV and TEST dataset individually to provide a baseline. For multi-task support, we implemented a task-specific, low-dimensional multi-head attention layer based on (Stickland and Murray, 2019)'s approach. The model demonstrated basic capability of multiple heads for sentiment analysis, paraphrase detection and textual similarity tasks.

4.3 Multi-head Architecture

Figure 2 shows an overview of the architecture. Building on top of the BERT base model, we added three different multi task heads with corresponding evaluation logics for finetuning.

The of BERT architecture makes it unsuitable for semantic similarity search or unsupervised tasks like clustering. Sentence Bert (Reimers and Gurevych, 2019) demonstrated a modified BERT with siamese and triplet network structures that can be compared using cosine-similarity. It shows great performance on Semantic Textual Similarity Benchmark(STS-B). Inspired by SBERT's work, we extracted the pooling logic from SBERT [1] and implemented with our Multi-Task model for the sentence embedding processing. The goal is to improve performance for paraphrase and Semantic evaluation. On top of the default CLS-TOKEN, there are two additional pooling strategies added:

- MEAN strategy: compute the average of all contextualized word embedding produced by BERT. This strategy provides a fixed dimensional output vector regardless of the input text length.
- MAX strategy: take a max-over-time of the BERT output vectors. It captures the most salient features across the entire sentence

4.4 Task Specific heads

To capture the relation between sentences, we can either treat it as a classification problem or a regression problem. In this project, we considered both cosine-similarity and Euclidean distances as similarity measures and compare between sentence embeddings.

4.4.1 Sentiment Analysis Task

The first task is Sentiment Analysis. The goal of Sentiment Analysis in Natural Language Processing (NLP) is to identify, extract, quantify, and study affective states and subjective information from text data. This task should analyze a sentence and suggest what is the sentiment category it's matching to. In this project, we are working towards a general version defined by Stanford Sentiment Treebank. (Socher et al., 2013) It defines five sentiment labels from Negative to Positive.

This is a classification problem. We use one linear layer on top BERT Layer with a soft-max function to find the output with highest possibility. For multi-class classification, where there are M=5 classes, we calculate a Cross Entropy loss based on each class label per observation:

$$\text{CrossEntropy\_Loss} = -\sum_{c=1}^{M} y_{x,c}\log(p_{x,c})$$

Where: (y) represents the ground truth label (0 or 1 for binary classification, one-hot encoded vector for multi-class).

[1] https://github.com/UKPLab/sentence-transformers/

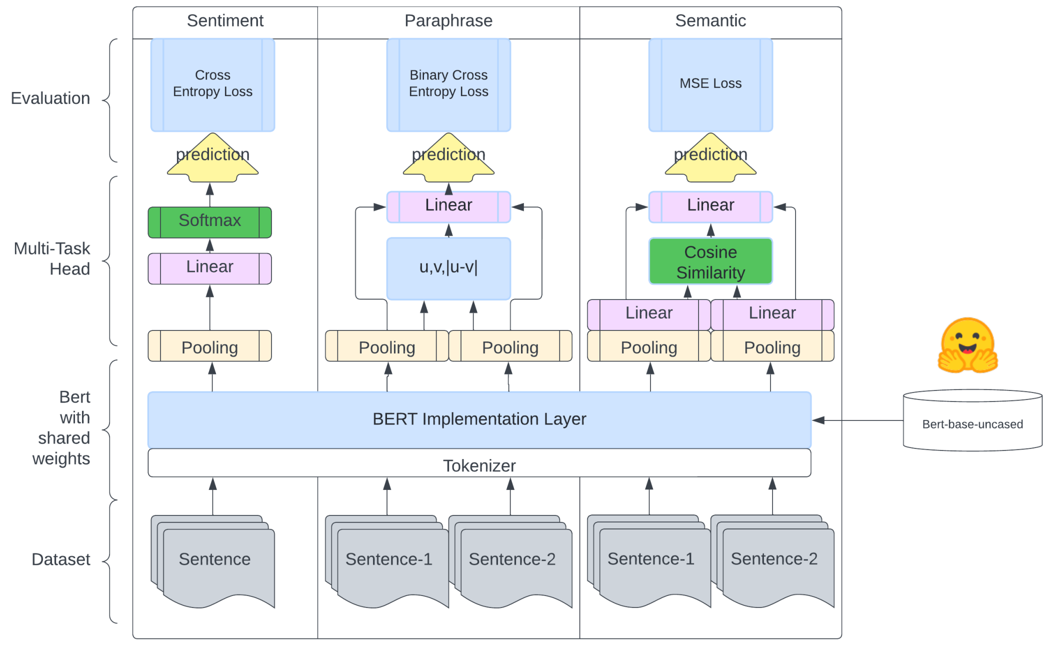

Figure 2: Project Model Architecture Diagram

4.4.2 Paraphrase Task

Paraphrase task is to provide a boolean suggestion on whether two sentences have the same meaning. BERT is pre-trained on masking tokens from the word embedding in the sentence. So pre-processing on the sentence doesn't yield good result on pre-trained weights. Based on (Reimers and Gurevych, 2019)'s analysis on multiple options in calculating the distance, we consider the Euclidean distances as similarity measures and compare between sentence embedding. Classification Objective Function by concatenate the sentence embedding u and v with element-wise difference $|(u - v)|$ and multiply it with the trainable weight. Then we optimize for the cross-entropy lose:

$$\text{BCE\_Loss}(y, p) = -\left(y \log(p) + (1 - y) \log(1 - p)\right)$$

Where: (y) represents the ground truth label (0 or 1 for binary classification)

Note depending on the loss calculation, different tasks may produce loss value at different scale. We adopted similar loss function to weight all loss equally between 0-1 range.

4.4.3 Semantic Task

Semantic Textual Similarity is a measure of semantic equivalence between a pair of text segments. In NLP, STS quantifies the degrees of similarity or relatedness in meaning between two sentences. This task is usually treated as a regression problem. State of the art methods often learn a complex regression function that maps sentence bedding to a similarity score. We adopt a linear to adjust the weight before calculating the cos-sim and concatenate with sentence embeddings for prediction.

Cosine similarity is defined

$$\text{Cosine similarity} = \frac{\mathbf{A} \cdot \mathbf{B}}{|\mathbf{A}| \cdot |\mathbf{B}|}$$

where $|\mathbf{A}|$ and $|\mathbf{B}|$ denote the Euclidean norms (magnitudes) of the vectors.

The Loss is measure with Mean Squared Error:

$$\text{MSE\_Loss} = \frac{1}{mn} \sum_{i=1}^{m} \sum_{j=1}^{n} (x_{ij} - y_{ij})^2$$

4.5 Projected attention layers

PAL and Adapters are two common ways for building multi-task BERT model with pretrained weights and they have proven to be effective. In this practice, we also build a PAL version based on our BERT implementation and try to evaluate the performance with the same multi-task heads design. Our PAL implementation is inspired by works by JosselinSomervilleRoberts [2] and rewritten for comparison purpose. The core idea of PAL is by introducing a low-rank task specific attention layer in parallel to the BERT self-attention, we will be able to share the weights of BERT model while still be able to store some parameters optimized for each tasks.

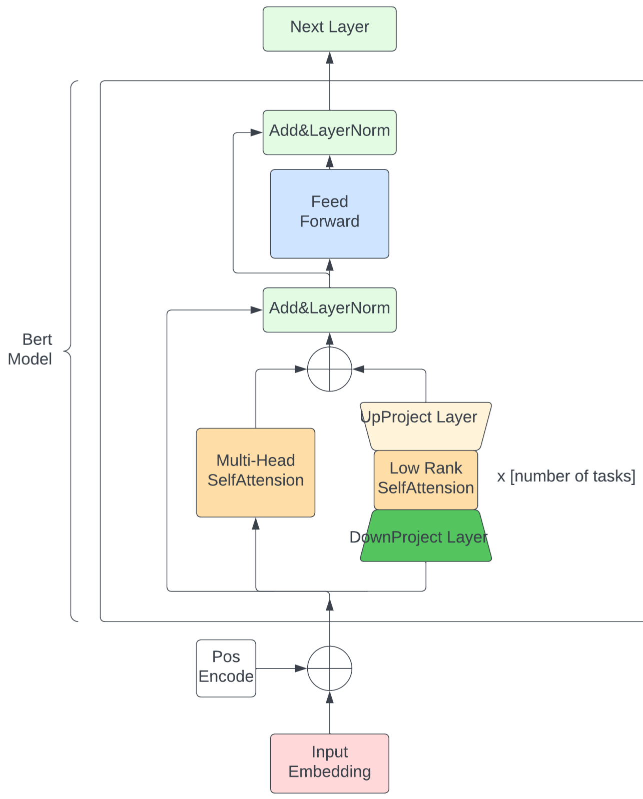

Figure 3: PAL Implementation

4.6 Details

The training setup is locally in RTX GPU with 24GB RAM. Batch size is fixed at 16 to accommodate memory limit.

[2]https://github.com/JosselinSomervilleRoberts/BERT-Multitask-learning

For **Baseline model**, we adopted a learning rate of 1e-05, hidden dropout probability of 0.3. We train 10 epochs. For finetuning on **non-baseline** run, we adopt a initiate learning rate of 2e-5.

AdamW optimizer is set as typical betas with $\beta_1 = 0.9$ and $\beta_2 = 0.99$, $\epsilon = 10^{-6}$. Bias correction is applied but no weight decay for the baseline. For the run other than baseline, a learning rate scheduler is integrated. There are four schedulers available: constant$_{s}chedule_{w}ith_{w}armup, linear_{s}chedule_{w}ith_{w}armup, cosine_{s}chedule_{w}ith_{w}armup, polynomial_{d}ecay_{s}chedule_{w}it$

4.7 Data

Datasets:In the experiment, we used the following dataset provided from default final project: SST-5, QQP and STS-B.

- **Stanford Sentiment Treebank**: A dataset consists of 11855 single sentences extracted from movie reviews and parsed by Stanford parser as 215154 unique phrases. It comes with sentimental label annotation on a scale from 0 (negative), 1(somewhat negative), 2(neutral), 3(somewhat positive), 4(positive).(Socher et al., 2013)
- **Quora Question Pairs**(QQP): A subset of the Quora dataset, consists of question pairs with labels indicating whether particular question is a paraphrase of another. The expected binary value shows whether the two sentences are equal or not. A random value may provide around 0.5 accuracy, which we can use as a bottom line.
- **SemEval STS Benchmark**: A dataset consists of 8628 different sentence pairs with similarity label on a scale from 0(unrelated) to 5(equivalent meaning). (Cer et al., 2017)

4.8 Evaluation method

- **SST**,**QQP** and **STS** are used for finetuning purpose and benchmark. For **SST** and**QQP**, we finetune the model on the DEV set and assess the accuracy on the TEST set based on whether the result matches the original label as the ground truth. For **SST**, we will predict the similarity values and evaluate with Person correlation with true similarity values.

5 Results

5.1 BERT finetune for each task

BERT Single Task Finetune	
Method	DEV ACC
STS finetuned	0.528
QQP finetuned	0.838
STS-B Finetuned	0.421

Table 1: Results of BERT Finetuned for Single Tasks

The above results are generated from the finetuned BERT model with task specific head on EVERY SINGLE dataset solely. By loading the Pretrain weights from Huggingface, we are able to achieve moderate results on three tasks. Note they are all based on a learning rate of 1e-5 with Adam optimizer. Batch size is 16 with Epoches of 10.

5.2 Multi Task BERT result

Accuracy			
Method	SST DEV	QQP DEV	STS-B DEV
SST-Finetuned BERT	0.528	0.510	0.283
MultiTask BERT	0.480	0.835	0.679
SBert(MAX,MAX,MAX)	0.358	0.742	0.630
SBert(MEAN,MEAN,MEAN)	0.342	0.826	0.657
SBert(CLS,MEAN,MEAN)	0.500	0.866	0.767
Original PALs	0.491	0.869	0.580

Table 2: Comparison of Multi-Task BERT Models

Some highlights of the Multi-Task BERT Model:

- The first SSS-Finetuned BERT is a baseline which was solely finetuned on SST dataset and evaluated against QQP and STS-B
- A MultiTask BERT is finetuned on three datasets to achieve an overall lower Loss.

$$Loss_{\text{overall}} = Loss_a + Loss_b + Loss_c$$

- SBert is BERT model with pooling enabled on the logits in forward function. In the case of SBERT (MAX, MAX, MAX), it means MAX-pooling is enabled for all three tasks, while SBERT (CLS, MEAN, MEAN) means no pooling for sentimental analysis but MEAN-pooling for Paraphrase and Semantic analysis tasks.
- Original PALs is BERT model with Projected attention layers. There three tasks specific low rank attention weights trained and stored. The low rank size is chosen to be 60, which represents a small parameter size.

6 Analysis

With the first STS-Finetuned BERT as the baseline, we can see finetuning on SST dataset can fit on the sentence sentimental information but is generated little value in completing Paraphrase and Semantic analysis tasks. Their optimization objectives are not aligned.

The best overall performance of Multi-Task BERT model is with the following setup: CLS-Token (No Pooling) for Sentimental and MEAN pooling for Paraphrase and Semantic Analysis. With SST dev accuracy: 0.500 Paraphrase dev accuracy: 0.866 STS dev correlation: 0.767 In comparison with baseline of individual tasks, we can see: 1. the SST performance decreased from 0.528. This is probably due to QQP and STS-B have different optimization objective compared with SST, which caused a conflict in gradient direction. Increasing the Sentiment training set size would help in this case. 2. The QQP and STS-B performance both are improved in comparison of BERT tuned for individual tasks. That means the QQP and STS-B dataset are contributing to each other in training, which helps to extract better common understanding in the sentence tokens. 3. The QQP and STS-B Performance also are improved in comparison of the MultiTask finetuned BERT model. SBERT MEAN Pooling would be the reason in this case as this is the common contributing factor for both variations.

When digging into SBert variations and compare them with the Baseline, we can see SBert Pooling of MAX and MEAN strategies both show performance improvement over the QQP and STS-B dataset, in comparison with default CLS-Token. However, SST task still runs best on the CLS-Token. MEAN and MAX pooling strategy is not helping in sentimental analysis. This observation is aligned with the finding in original Sentence Bert paper(Reimers and Gurevych, 2019).

Paraphrase binary classification problem is relative easy to train since we are provided with a relatively larger dataset. In average, it took only 3 epoches to reach a relatively good accuracy rate(>0.85).

Complex structure may achieve good training accuracy but is prone to overfitting issue. It may learn more irrelevant context which only works in the training dataset but not transferable to DEV and TEST dataset.

7 Future works

One future work is the analysis of the impact of SMART Regularization (Jiang et al., 2020). It is implemented for the project codebase but there were not enough data generated to evaluate the impact on the result. I am expecting a good adjustment on the ratio would help the overfitting on problem.

We also integrated one version with idea of BITNET (Ma et al., 2024). The work tries to use bitlinear layer of only [-1,0,1] values to train on the same tasks. Our initial attempt didn't yield good results so we are not including them into the report. We will continue exploring the possibilities of completing the works with less complex model with similar performance.

8 Conclusion

Our implementation demonstrates the potential of BERT to adapt to multiple tasks simultaneously, leveraging shared representations while maintaining task-specific capabilities through strategic architectural modifications like Projected Attention Layers (PALs) and SBERT pooling strategies. Our project underscores the delicate balance required in multitask model training, where task-specific optimizations must be carefully weighed against the overarching model coherence to prevent performance degradation in any single task.

References

Quora dataset release question pairs.

Qiwei Bi, Jian Li, Lifeng Shang, Xin Jiang, Qun Liu, and Hanfang Yang. 2022. MTRec: Multi-task learning over BERT for news recommendation. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 2663–2669, Dublin, Ireland. Association for Computational Linguistics.

Daniel Cer, Mona Diab, Eneko Agirre, Inigo 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)*. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for nlp.

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*. Association for Computational Linguistics.

Xiaodong Liu, Pengcheng He, Weizhu Chen, and Jianfeng Gao. 2019. Multi-task deep neural networks for natural language understanding.

Shuming Ma, Hongyu Wang, Lingxiao Ma, Lei Wang, Wenhui Wang, Shaohan Huang, Li Dong, Ruiping Wang, Jilong Xue, and Furu Wei. 2024. The era of 1-bit llms: All large language models are in 1.58 bits.

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.

Jonas Pfeiffer, Andreas Rücklé, Clifton Poth, Aishwarya Kamath, Ivan Vulić, Sebastian Ruder, Kyunghyun Cho, and Iryna Gurevych. 2020. Adapterhub: A framework for adapting transformers.

Alec Radford and Karthik Narasimhan. 2018. Improving language understanding by generative pre-training.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks.

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.

Asa Cooper Stickland and Iain Murray. 2019. Bert and pals: Projected attention layers for efficient adaptation in multi-task learning.

Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. 2020. Gradient surgery for multi-task learning.