miniBert Unleashed

Stanford CS224N Default Project

John Cao
Department of Electrical Engineering
Stanford University
johncao@stanford.edu

David Kwentua
Department of Computer Science
Stanford University
dkwentua@stanford.edu

Abstract

This project aims to enhance miniBERT's performance on sentiment analysis (SST), paraphrase detection (PARA), and semantic textual similarity tasks (STS). Employing a multifaceted approach encompassing multi-task learning, random dataset sampling, and Siamese network architectures, we investigate avenues to augment miniBERT's efficacy. Through systematic experimentation, we uncover nuanced insights, revealing the efficacy of random sampling of datasets and mean pooling of embeddings in improving model performance, while unexpected downturns are observed with cosine similarity fine-tuning, particularly in STS tasks. Despite challenges, our proposed model architecture shows promising results, notably in Para and STS tasks.

1 Key Information to include

- Mentor: Arvind Mahankali
- External Collaborators: No
- Sharing project: No
- Team Contributions: Worked together on part 1 of the project, writing the project milestone, and the final report. For part 2, John implemented and ran the code and David ran inferences.

2 Introduction

The advent of Transformer-based models has revolutionized the field of Natural Language Processing, with Bidirectional Encoder Representations from Transformers (BERT) standing out as a particularly influential model. BERT's ability to generate rich word embeddings has enabled significant advancements across different NLP tasks. However, despite BERT's powerful capabilities, there exists the challenge of efficiently adapting it to multiple tasks. Complications such as disagreement between task objectives, varying data accessibility across tasks and effective transfer learning for adapting to task-specific domains make learning multiple tasks a difficult problem. This highlights the need for efficient multi-task approaches.

In this context, we explore the potential of a minimal BERT model to address the challenges associated with multi-task learning. Our investigation focuses on the following adaptations: A combined loss function which sums all task specific losses, random dataset sampling to balance training given imbalanced datasets, gradient surgery for aligning conflicting task objectives, Siamese network architecture to produce robust and rich vector representations of sentences and a convolutional neural network backbone to further refine the BERT embeddings. Through a series of experiments and methodical enhancements, we aim to augment miniBERT's efficacy in sentiment analysis, paraphrase detection, and semantic textual similarity tasks. This pursuit not only seeks to enhance task-specific performance but aims for better generalization across a spectrum of NLP tasks.

3 Related Work

The development of BERT (Devlin et al. (2019)) marked a significant advancement in NLP, setting new benchmarks across a range of tasks through its transformer-based architecture. BERT's success has spurred a plethora of research aimed at improving and adapting transformer models for various NLP applications. Multi-task learning has been explored extensively in NLP, aiming to improve model generalization by leveraging domain-specific information from related tasks. Works like MT-DNN **?** demonstrated the effectiveness of using a shared transformer encoder for multiple tasks, and highlight the potential of multi-task learning to improve model performance and generalization. Similarly, Raffel et al. (2023) unified multiple NLP tasks using a text-to-text framework, demonstrating significant gains in performance across diverse tasks.

Gradient conflicts are a constant challenge in multi-task learning, as gradients from different tasks can interfere with each other, resulting in suboptimal performance. Yu et al. (2020) proposed a method called PCGrad (Projected Conflicting Gradients), which projects the gradients in order to improve multi-task learning stability. This technique of ensuring harmonious gradient updates has shown significant promise in enhancing the performance of multi-task learning models. In our project, we explore the use of gradient surgery to address conflits that arise between the three aforementioned tasks.

Random Sampling methods have been applied extensively in multi-task learning to balance the learning process across tasks with varying dataset sizes. More recently, newer methods of addressing this problem have been brought forward; for instance, Wang et al. (2020) introduced an algorithm called MultiDDS that serves the aforementioned purpose, albeit in the context of Neural Machine Translation. We adopt a random subset sampling approach in this project to mitigate the impact of the size discrepancies between our datasets.

Siamese network architectures have proven effective for tasks requiring comparisons of two input sequences like semantic texual similarity (STS) and paraphrase detection. Reimers and Gurevych (2019) introduced Sentence-BERT, which used a Siamese network to derive meaningful sentence embeddings. This approach significantly improved the performance of sentence-pair tasks by providing a robust way to compute sentence similarity. Inspired by this work, we employ a Siamese network architecture for the STS and Paraphrase detection tasks.

The integration of convolutional neural networks (CNNs) with transformer models has been explored to enhance feature extraction from word embeddings. Kim (2014) showed that applying CNN to NLP tasks improved performance when combined with pretrained word vectors; this motivates our use of a CNN backbone to further refine the embeddings produced by miniBERT.

4 Approach

4.1 Baselines

Our baseline approach involves naively fine tuning miniBERT sequentially on the three downstream tasks: Sentiment Analysis (SST) -> Semantic Textual Similarity (STS) -> Paraphrase Detection (Para). The fine tuning was done on the full model as opposed to just the last linear layer. We do this for all our subsequent experiments unless explicitly stated otherwise. The sequential nature of this method is naturally prone to catastrophic forgetting, which is especially true in our case since we trained the Para dataset last, which is much larger than the STS and SST datasets. We noticed that the baseline accuracy gained from this approach for the sentiment analysis task is much lower than the baseline provided in the handout, only 0.220. So we decided to replace this part of the baseline with the number provided in the handout, while using our own results for the STS and Para tasks.

4.2 Multi-task Learning and Gradient Surgery

Our main approach to achieve multi-task learning is to compute a single loss consisting of the sum of three task-specific losses. More specifically, we define the loss

$$L_{\text{multi-task}} = L_{\text{Para}} + L_{\text{STS}} + L_{\text{SST}}. \tag{1}$$

We use the following losses: Binary Cross-Entropy Loss for paraphrase detection, Mean Squared Error loss for semantic textual similarity and Cross-Entropy loss for sentiment analysis. This formulation

of the loss function guides the optimizer to find the parameters which minimize all three individual loss functions, thus learning all tasks at the same time. However, it has been shown that multi-task loss formulations such as (1) are often very difficult to learn, and can lead to worse performance than single-task learning on the individual tasks Rusu et al. (2015); Parisotto et al. (2015). To this end, we use the method of *gradient surgery* presented by Yu et al. (2020). The aim of this method is to solve the problem of *conflicting gradients*, which describes the situation where the gradients of the different loss functions point in opposing directions (cosine similarity < 0), causing learning to stagnate in local minimas. Gradient surgery is defined as the operation

$$g_i = g_i - \frac{g_i \cdot g_j}{||g_j||^2} \cdot g_j, \tag{2}$$

where g_i is the gradient of the i-th task and g_j is a conflicting gradient. Equation (2) projects g_i onto the normal plane of g_j, thus removing the conflicting component.

4.3 Random Subset Sampling

We explored two different approaches for handling the enormous differences in size between the datasets, with the Quora dataset being around 33 times and 47 times larger than the SST and STS datasets respectively. The two approaches are as follows:

1. Train on the entire Quora dataset during each epoch. Each time the smaller datasets are exhausted, we recycle them until the entire epoch is finished. The batch sizes are kept the same for all tasks.
2. In each epoch, randomly sample a subset of the Quora dataset while massively increasing the batch size for the paraphrase detection task. This approach is motivated by the potential of overfitting from repeatedly recycling the smaller datasets while going through the Quora data set. The reduced size of the data set from subsampling, along with the increased batch size is aimed to result in fewer iterations of re-use of the SST and STS datasets, thus hopefully reduce the risk of overfitting.

4.4 Siamese Network Architecture, Pooling and Cosine Similarity Fine Tuning

For the semantic textual similarity and paraphrase detection task, we use a Siamese network architecture as proposed by Reimers and Gurevych (2019). For a sentece pair, we make two individual passes of each sentence through the BERT layers, producing a pair of embeddings. All embeddings, including the CLS embeddings, are then mean pooled and concatenated and combined into a single vector representation of the sentences. The intuition is to incorporate both the global sentence information from the CLS embedding with the local information from the individual word embeddings. Given two pooled n-dimensional embeddings $u, v \in \mathbb{R}^n$, we concatenate vectors along with their absolute distances into a single vector, and project it to a single logit using a linear layer. Mathematically, this can be written as

$$\text{Logits} = W(u, v, |u - v|) + b, \tag{3}$$

where $W \in \mathbb{R}^{1 \times 3n}$ is the weight matrix and b is the bias. Note that for paraphrase detection and sentiment textual similarity, the output logit is a single scalar, hence the singleton dimension of the rows of W. The model can be made more expressive by combining more liner layers, which we will investigate in our experiments. For sentiment analysis, we also apply mean pooling of the embeddings, before projecting the output to 5 logits using a linear layer, representing the 5 sentiment scores.

We also experiment with cosine similarity fine tuning for the semantic textual similarity (STS) task. Given pooled embeddings u, v, the cosine similarity between them is defined as

$$\text{Cosine Similarity} = \frac{u \cdot v}{||u||_2 ||v||_2}. \tag{4}$$

This value is then used in place of the logit produced by the linear projection layer. The intuition is that similar paraphrases will admit similar vector representations, therefore a high cosine similarity. Fine tuning using this metric therefore encourages the model to learn similar embeddings for sentences which are similar to each other.

4.5 Convolutional Backbone for Classification Heads

For each task, we implement a 1D convolutional layer as a backbone for the final MLP layers, taking the output of BERT as an input. This approach is inspired by the work of Kim (2014), which investigated the application of convolutional neural networks for NLP tasks, using them as an additional feature extractor for the learned word embeddings. It was shown the in paper that the addition of a convolutional layer improved the performance of the model in several NLP domains when combined with pretrained word vectors, including the task of sentiment classification trained on the Stanford Sentiment Treebank dataset. We hypothesize that the addition of a convolutional feature extractor to our BERT embeddings would yield similar performance gains by learning a set of filters to extract the most essential information from the embeddings. To add further flexibility to our architecture, we also add a residual connection which bypasses the CNN layer. This is to ensure that the model performs at least on par with a simpler variant which does not have the CNN, and is able to utilize the expressivity of the CNN if it indeed is able to provide performance improvements. An illustration of our full model architecture is shown in figure 1.

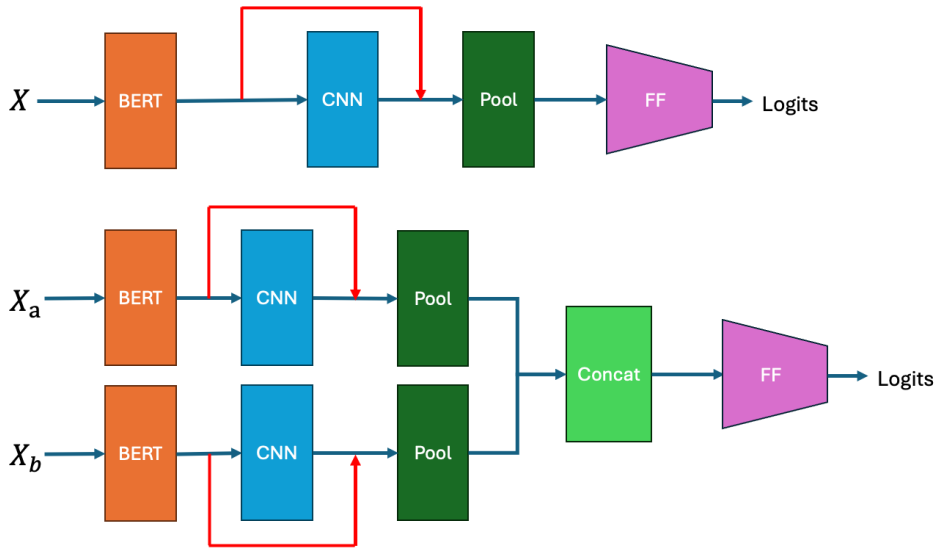

Figure 1: Our model architecture. The top part illustrates the model for the sentiment classification task. The bottom part illustrates the model for the paraphrase detection and sentiment textual similarity task. The red lines represent residual connections.

5 Experiments

5.1 Data

We used the three datasets provided for the default project for our experiments:

1. **Stanford Sentiment Treebank**: The Stanford Sentiment Treebank (SST) dataset Socher et al. (2013) is composed of 11855 single sentences from movie reviews, each given a sentiment score between 0 and 4. The dataset is split up between 8544 train, 1101 dev and 2201 test examples.

2. **SemEval STS Benchmark:** The SemEval STS Benchmark dataset Agirre et al. (2013) consists of 8628 different sentence pairs of varying similarity rated from 0 (unrelated) to 5 (equivalent meaning). The dataset is split between 6040 train, 863 dev and 1725 test examples.

Method	Abbreviation
Simple recycling (No.1 in 4.3)	SR
Random subset sampling (No.2 in 4.3)	RSS
Cosine similarity	CS
Mean pooling	MP
Max time pooling	MTP
Gradient surgery	GS
Convolutional neural network	CNN
Residual connection	Res
No residual connection	NoRes
Additional n output MLP layers	MLP[n]

Table 1: Abbreviation conventions used in the paper.

3. **Quora Dataset:** The Quora dataset [1] consists of 404,298 question pairs with labels indicating whether they are paraphrases of one another. The dataset is split between 283101 train, 40429 dev and 80859 test examples.

5.2 Evaluation method

We evaluate the performance of our models using the following metrics:

- **Sentiment Analysis:** The classification accuracy - the proportion of correctly classified examples out of the total.
- **Paraphrase Detection:** The classification accuracy - the proportion of correctly identified paraphrase pairs.
- **Semantic Textual Similarity:** Pearson Correlation Coefficient (measures the linear correlation between the predicted similarity scores and the ground truth scores).

5.3 Abbreviations

We abbreviate our methods for the sake of brevity. Table 1 summarizes our conventions.

5.4 Experimental details

All the training were conducted on an 80 GB H100 GPU over 10 epochs with a learning rate of 10^{-5}, which we found to work the best from trying a variety of configurations. In each epoch, we randomly sample 140000 datapoints from the Quora dataset with a batch size of 128, while using a batch size of 8 for the STS and SST datasets. For the CNN, we used a kernel size of 3 and a number of filters equal to the embedding size, which is 768. The hidden layers of the output projection network were configured to project the input down to an intermediate hidden dimension of 384. Dropout was applied to the last hidden layer with a dropout probability of 0.3.

5.5 Results

Our results on the dev set are summarized in table 2. Our test leaderboard results are given in table 3. We conducted our experiments by adding the proposed methods outlined in section 4 one at a time, with the purpose of gauging their individual impact on the performance.

The simple recycling of the smaller datasets (SR) without any other modifications other than introducing the multi-task loss yielded slightly higher overall score than the baseline. As predicted, switching to the random sampling (RSS) of the Quora dataset seem to significantly boost the performance of the model for the Para and STS tasks, resulting in a 0.041 increase in the overall score. An even bigger performance improvement was attained by using mean pooling of all the output BERT embeddings instead of relying soley on the CLS embedding (RSS + MP). Surprisingly, introducing cosine similarity fine-tuning (RSS + MP + CS) for STS gave lower scores for all metrics. Due to

[1] `https://www.quora.com/q/quoradata/First-Quora-Dataset-Release-Question-Pairs`

this observation, we decided to continue our subsequent experiments without cosine-similarity fine tuning and rely soley on mean pooling across all task heads. The addition of gradient surgery (RSS + MP + GS) led to a slight increase in the scores for paraphrase detection compared to all previous models, and an increase in the SST accuracy compared to the model without cosine-similarity.

To investigate the effectiveness of the CNN approach suggested by Kim (2014) for multi-task BERT, we began by implementing the CNN backbones without any residual connections (RSS + MP + GS + CNN + NoRes) to remain faithful to the original implementation. Interestingly, the model exhibited worse performance across all tasks after introducing the CNN contrary to our expectations. Even after trying different configurations for the kernel size and the number of filters, we did not observe any meaningful improvements to the performance. In Kim (2014), the author proposed to use max time pooling instead of mean pooling. Making the switch to this approach (RSS + MTP + GS + CNN + NoRes) did not yield any drastic changed either, only resulting in a slightly lower overall score. We further experimented with adding an addition hidden layer to the output projection network while keeping the previous configurations (RSS + MP + GS + CNN + NoRes + MLP[1]). This simple adjustment resulted in a 4.5% increase in the overall score of the model. We then added the residual connection around the CNN backbone as described in section 4.5 (RSS + MP + GS + CNN + Res + MLP[1]). This addition further enhanced the overall score of the model and yielded the best paraphrase detection accuracy out of all of our models. Lastly, we added one additional hidden layer to the projection layers (RSS + MP + GS + CNN + Res + MLP[2]), which further added 0.08 points to the overall score, making it our highest overall performing model and the best one for SST and STS.

Method	**SST**	**Para**	**STS**	Overall
Baseline	0.515 (Handout)	0.632	0.304	0.600
SR	0.507	0.633	0.362	0.607
RSS	0.500	0.759	0.371	0.648
RSS + MP	0.502	0.832	0.817	0.747
RSS + MP + CS	0.493	0.791	0.433	0.669
RSS + MP + GS	0.510	0.842	0.818	0.754
RSS + MP + GS + CNN + NoRes	0.491	0.787	0.715	0.712
RSS + MTP + GS + CNN + NoRes	0.497	0.782	0.684	0.707
RSS + MP + GS + CNN + NoRes + MLP[1]	0.518	0.858	0.680	0.739
RSS + MP + GS + CNN + Res + MLP[1]	0.486	**0.867**	0.855	0.760
RSS + MP + GS + CNN + Res + MLP[2]	**0.520**	0.848	**0.870**	**0.768**

Table 2: Experimental results on the dev test set.

Method	**SST**	**Para**	**STS**	Overall
RSS + MP + GS + CNN + Res + MLP[2]	0.533	0.847	0.859	0.770

Table 3: Test leaderboard results.

6 Analysis

6.1 Impact of Random Subset Sampling

It was evident during training that the simple recycling of the smaller datasets led to aggressive over-fitting, as the train accuracies kept increasing while the dev scores quickly stagnated. This is not surprising since a single epoch for the Quora dataset is equivalent to around 40 epochs for the other datasets using a batch size of 8, with each epoch taking around 35000 iterations. Using our configuration of sampling 140000 datapoints in each epoch and a batch size of 128, each epoch now takes around only 1094 iterations to complete. This results in the model training on the SST dataset only once per epoch, and roughly 1.5 times for the STS dataset. We generally noticed a positive correlation between the sample size, batch size and the model performance. However, the memory constraints of the GPUs limited us from going beyond our current configuration. Overall, randomly

sampling a subset of the Quora dataset and training the model with a larger batch size had the desired effect of improving generalization while simultaneously reducing the time required to train the model.

6.2 Impact of Cosine Similarity Fine Tuning and Mean Pooling

Using mean pooling of all the embeddings produced by the Siamese BERT and concatenating them with the absolute distance between the pooled embeddings greatly improved the performance of our model over only using the CLS embeddings. This result aligns with the findings of Reimers and Gurevych (2019). As mentioned in the article, the inclusion of the absolute distances formation between two embeddings can help the model distingush between similar sentences or if two sentences are paraphrases of each other, with the intuition being that sentences that are alike should be close to each other in embedding space. We hypothesize that the mean pooling operation improves performance by enhancing the global information in the CLS embedding with the local contextual information contained in each token embedding, leading to a more hollistic representation of the whole sentence. Furthermore, mean pooling might also aid in averaging out the any noise present in the data, making the model more robust to overfitting and outliers.

Interestingly, using cosine-similarity fine tuning drastically worsened the performance of the model on the STS task instead of improving it. Our theory as to why this might be the case relates to Reimers and Gurevych (2019), which mentions that the Siamese network design along with mean pooling and absolute distances as input features makes the sentence embeddings comparable with cosine similarity. If this is indeed the case, then normal fine tuning already achieves what cosine similarity aims to do, i.e make similar pairs of sentences/paraphrases similar in embedding space. This means that removing the MLP layer might just equate to removing the expressivity it inherently possesses, replacing it with a mechanism which was already there in the first place.

6.3 Impact of CNN Backbone

The implementation of the CNN backbone architecture worsened the overall performance of the model, as can be seen in table 2. Even with max time pooling as was used in Kim (2014), the results still largely remained the same. One possible explanation could be caused by the well-known difficulty of learning identity mappings for neural networks, the same phenomenon which motivated the ResNet architecture He et al. (2016). Introducing an additional convolutional layer might not help if the input embeddings are inherently rich in relevant information, making it difficult to extract more. For our case, this might very well be the case considering that BERT is built using several layers of multi-headed attention blocks, enabling it to learn embeddings which contain extremely rich contextual information. In comparison, Kim (2014) used word2vec embeddings Mikolov et al. (2013) which are produced without the attention mechanism, therefore lacking the local and global contextual information which the BERT embeddings posses. This leaves more room for potential improvements to be found by a CNN. Inspired by ResNets, we added residual connections to route the input to the output of the CNN to facilitate the learning of the identity mapping when needed, allowing the model to choose wether or not to utilize the CNN for its predictions. The improved scores we obtained from adding the residual connection supports our hypothesis, that the addition of the CNN layer in this context is in fact redundant due to the rich semantic information already present in the original BERT embeddings. Lastly, adding subsequent hidden projection layers improved the performance as expected, and demonstrates the empirically observed negative correlation between the number of parameters and test loss within deep learning, as described by Kaplan et al. (2020).

7 Conclusion

Our experiments unfolded a series of noteworthy findings. Firstly, we observed that simple recycling of smaller datasets leads to overfitting, highlighting the importance of RSS to compensate for dataset size discrepancies. Additionally, mean pooling of all embeddings, combined with the absolute distances between pooled embeddings, significantly boosted model performance, which aligns with prior research findings. On the contrary, the introduction of CS fine-tuning led to a decline in our model's performance across all tasks 2, with a near 50% drop in STS correlation. Furthermore, integrating a CNN backbone failed to yield improvements. This could possibly be due to the rich contextual information already present in the BERT embeddings, effectively rendering additional

feature extraction redundant at best, and detrimental at worst. Despite these challenges, our model architecture achieved promising results, especially in paraphrase detection (0.848) and STS (0.870).

In conclusion, this project underscores the complexity of enhancing miniBERT's performance. While certain techniques like GS, RSS, and MP yielded substantial improvements, others, like CNN, posed unexpected challenges. Moving forward, avenues for future work include additional pretrainign of miniBERT to further enhance its ability to capture nuanced linguistic patterns. Additional considerations for future work include refining the integration of CNN backbones and investigating scaling laws within the multi-task learning framework.

8 Ethics Statement

The development and deployment of NLP models like miniBERT, and the methodologies employed to enhance their performance, raise several ethical considerations which are crucial to address and mitigate. One primary ethical consideration is the potential for biases inherent in the datasets used for training and evaluation. Biases in data can lead to biased model predictions, perpetuating, and even exacerbating, societal inequalities. Mitigation strategies for this concern include thorough data preprocessing to identify potential biases and diversifying training datasets to ensure representation from various demographics.

Another ethical challenge is the responsible use of NLP models in sensitive applications such as sentiment analysis and paraphrase detection. Biased or inaccurate predictions in these tasks can have significant real-world consequences, including misinformation propagation and discrimination. Mitigation strategies may include incorporating transparency measures to clarify the limitations of model predictions, and implementing robust evaluation frameworks to assess model performance across diverse demographic groups.

Additionally, the computational resources required for training and deploying large-scale NLP models pose environmental concerns. The carbon footprint associated with training deep learning models and the disparity in access to computational resources can exacerbate existing environmental and socioeconomic inequalities. Mitigation strategies for this concern include exploring energy-efficient training methods and considering the environmental impact when designing a model.

Ultimately, while models like our miniBERT offer immense potential for positive societal impact, they also present ethical challenges that must be addressed proactively.

References

Eneko Agirre, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, and Weiwei Guo. 2013. * sem 2013 shared task: Semantic textual similarity. In *Second joint conference on lexical and computational semantics (* SEM), volume 1: proceedings of the Main conference and the shared task: semantic textual similarity*, pages 32–43.

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

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751, Doha, Qatar. Association for Computational Linguistics.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Emilio Parisotto, Jimmy Lei Ba, and Ruslan Salakhutdinov. 2015. Actor-mimic: Deep multitask and transfer reinforcement learning. *arXiv preprint arXiv:1511.06342*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2023. Exploring the limits of transfer learning with a unified text-to-text transformer.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*.

Andrei A Rusu, Sergio Gomez Colmenarejo, Caglar Gulcehre, Guillaume Desjardins, James Kirkpatrick, Razvan Pascanu, Volodymyr Mnih, Koray Kavukcuoglu, and Raia Hadsell. 2015. Policy distillation. *arXiv preprint arXiv:1511.06295*.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y 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.

Xinyi Wang, Yulia Tsvetkov, and Graham Neubig. 2020. Balancing training for multilingual neural machine translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8526–8537, Online. Association for Computational Linguistics.

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