Conclusion

5.4 Summary

This thesis has explored various ways to utilize BERT with adapters in machine translation. We start by understanding the impact of pre-training data in different domains and the contribution of different volumes in the pre-training data. We continue the study by leveraging different techniques to understand the impact of the pre-trained models by shuffling the pre-trained BERT weights and using randomly set weights. We then conduct the experiments to understand the importance of adapters in either the encoder or the decoder by showing the performance of adapters when they are removed from either of the components. Finally, we perform reduction experiments where we reduce the size of BERT by manually removing the weights either by zeroing out some of the values in the matrices or completely deleting them.

The experiments in Chapter 4 show that with the proper initialization, adapters can help achieve better performance than training the models from scratch while training far fewer weights than the original model. We further show that even with random fixed weights in the main part of the model, the adapters and cross-attention can recover and achieve performance similar to one of the baseline models.

In the subsequent experiments in Chapter 5, we find that fine-tuning adapters on the encoder side is more important than in the decoder. We also see a similar behaviour when we use the original BERT weights only on the encoder or

the decoder and fixed random weights on the other part. Interestingly, when the adapters were injected only to the decoder, with the encoder pre-trained or random, the performance dropped to zero. In other words, a fitting encoder is critical.

We further studied the behaviour of adapters when we tried to down-scale the pre-trained model size. In our experiments, we found that a model with just half of the weights, such as our `zsbert`, can closely match the performance of the baseline model, the model that uses BERT in both the encoder and the decoder and only fine-tuning the cross-attention and output layers. Finally, we also observe that we can increase further the effectiveness of adapters in `zsbert` by only incorporating adapters on the encoder side.

We see two practical applications from our findings:

- Initializing just the encoder with the pre-trained weights such as BERT (with a fixed random decoder) and fine-tuning with adapters could be helpful when targeting low-resource languages. The random decoder is created trivially, and no large target side monolingual corpus is needed.

- Reducing the pre-trained BERT to half its size and fine-tuning with adapters provide useful GPU memory savings while keeping a similar performance as the baseline model.

In summary, our experiments show the potential of adapters in the machine translation setup. We understand from the experiments that fine-tuning adapters with randomly set weights in the base pre-trained network can achieve similar results as training the entire transformer model with BERT configuration. Furthermore, the down-scaling experiments also show that with a random weight reduction technique, we can reduce the size of BERT and achieve similar performance as the BERT model that was fine-tuned by only modifying the cross-attention layer.

5.5 Future Works

In this thesis, we have seen the behaviour of adapters during fine-tuning on various scenarios where we modify the pre-trained models to the point where we were using a completely random value. It is interesting for us to see that the final performance of the model could sometimes achieve comparable quality as if we trained the whole BERT model from scratch. Further experiments would be interesting to solidify the results found in this thesis. We propose experiments with various model architectures to see whether we can still see the same behaviour. The goal of the experiments is to measure any correlations between the adapters and the architecture chosen for the base model. Until recently, most works in adapters only focus on transformer-based models. It would be interesting to perform similar experiments with LSTM-based models.

There are some works such as [6, 66] where they can identify the location of certain features and pieces of information within the pre-trained models. In this thesis, we have shown that when we remove arbitrary weights from BERT and later perform the fine-tuning with adapters, the model can perform similarly to the model that uses the whole BERT weights and only modify the cross-attention and output layers. This experiment shows that some lost information can be recovered in the adapters during fine-tuning as long as the adapters have enough capacity. It would be interesting if we could find a way to store a particular group of features by training the models modularly with adapters. [55] has shown that by incorporating adapters during the pre-training and using them as a module that stores the knowledge for each language, they can maintain a constant size of the model while increasing the number of languages that they feed to the model. It would be interesting to perform experiments where the adapters can store more granular linguistic features such as syntax and morphology-level features.