Adapters for Resource-Efficient Deployment of Intent Detection Models

Anonymous ACL submission

Abstract

Modern Transformer-based language models such as BERT are huge and expensive to deploy in practical applications. In environments such as commercial chatbot-as-a-service platforms that deploy many NLP models in parallel, less powerful models with a smaller number of parameters are often used to keep deployment costs down. Also, in times of climate change and scarce resources, the deployment of many huge models is no longer adequate. This paper proposes the BERT+Adapter architecture for hosting many models and saving considerable amounts of (GPU) memory. We further demonstrate the effectiveness of this approach using the example of intent detection for dialog systems. Many task-specific adapters can share one large Transformer model with the adapter framework. To deploy 100 NLU models requires 1 GB of memory for the proposed BERT+Adapter architecture, compared to 41.78 GB for a BERT-only architecture. Also, we show that the training time of the BERT+Adapter architecture is on average 14.43 times shorter than that of vanilla BERT. Furthermore, we demonstrate that the accuracy of BERT+Adapter on intent detection tasks is on par with a vanilla BERT architecture.

1 Introduction

Natural Language Understanding (NLU) is an essential component of dialog systems (DS). The NLU converts an unstructured user utterance into structured information: It comprises a) Intent Detection (ID) where the dialog system classifies a user utterance into a predefined list of intents. The system can understand what the user wants to say based on this classification. The other part of NLU is b) slot filling / entity recognition (ER) in which the dialog system fills in specific slots that belong to an intent.

Transformer-based models (Vaswani et al.) currently show the best results in ID (Mehri and Eric, 2021). Transformers models are very large, a standard BERT model (Devlin et al., 2019) for ID on the HWU64 dataset (Liu et al., 2019) occupies 417.75 MB of data in the Hugging Face implementation (Wolf et al., 2020). Hosting these large models is expensive and uses large amounts of computational power which is no longer adequate in times of climate change and scarce resources (Strubell et al., 2019). Therefore, smaller architectures were proposed such as DIET (Bunk et al., 2020) or ConveRT (Henderson et al., 2020) which use substantially less memory (63 MB for DIET in our HWU64 example). While state-of-the-art Transformer-based NLU models reach performances up to 92% accuracy (Mehri and Eric, 2021), practical NLU models show a weaker performance of 86-89% (Liu et al., 2019).

In this paper, we apply adapters (Rebuffi et al., 2017; Houlsby et al., 2019) to ID. We use a general-purpose pre-trained BERT model and introduce a small number of additional parameters (5.87 MB for our HWU64 example in the AdapterHub implementation (Pfeiffer et al., 2020)). During training, we freeze the parameters of the original BERT model and train only the additional parameters. Using this approach, we propose a resource-efficient method to deploy multiple ID models: Instead of deploying multiple BERT models, we only need a single, shared BERT model and one adapter for each downstream NLU application. The deployment of, for example, 100 ID models using this new framework requires 1 GB (417.75 MB for the BERT model + 100×5.87 MB for the adapters), instead of 41.78 GB (100×417.75 MB size of the BERT model).

Adapters show their strength in environments where many models are deployed in parallel. We demonstrate our approach in two areas of applications. First, we propose its use in chatbot-as-a-service platforms such as IBM Watson Assistant[1],

[1] https://www.ibm.com/products/watson-assistant

Google Dialogflow[2], and SAP Conversational AI[3]. These systems offer chatbots as a service and therefore host many DS and a large number of NLU models. This industry can save significant amounts of costs, resources, and energy using our approach.

The second application area is modular dialog systems (MDS) (Nehring and Ahmed, 2021) in which a DS is composed of several sub-DS. Often, each sub-DS uses its own NLU model and can benefit from the adapter approach's resource savings.

We perform experiments on three datasets to compare the performance of the BERT+Adapter model to a fully fine-tuned, vanilla BERT model. We determine the sizes of the models and show that for a large number of models, BERT+Adapter saves a significant amount of memory, compared to BERT and to the DIET (Bunk et al., 2020) model. The experiments show that the accuracy of ID of BERT+Adapter is comparable to BERT, despite the memory savings of BERT+Adapter. Also, we show that BERT+Adapter is trained considerably faster than BERT.

We publish the source code to reproduce our experiments on GitHub[4].

2 Background

2.1 Natural Language Understanding

The recent increase in research on dialog systems was a catalyst for research in NLU. Systems like the Dual Intent and Entity Transformer (DIET) (Bunk et al., 2020) or the Dual Sentence Encoders (Henderson et al., 2020; Cer et al., 2018) focus on lightweight models. DIET model uses pre-trained word embeddings like BERT or ConveRT as dense features. It combines these features with character n-grams and passes them through a Transformer with two instead of the usual 12 encoder layers. Smaller models are competitive with large-scale models in terms of performance and are much faster in training and inference. Full-size Transformer models are used for NLU also and achieve a stronger performance than the efficient architectures (Mehri et al., 2020; Mehri and Eric, 2021).

[2]https://cloud.google.com/dialogflow

[3]https://cai.tools.sap

[4]GitHub link available with camera-ready version. Until then, please download the source codes from https://bit.ly/3ItjPd2

2.2 Adapters

Instead of replacing the original model for a smaller one, adapters (Rebuffi et al., 2017; Houlsby et al., 2019) are a lightweight addition to Transformer models. An adapter is a small set of parameters inserted between the original model's layers, usually a pre-trained transformer model. For a text classification task as in our paper, the adapter also adds a classification layer on top of the original model. During training, the parameters of the original Transformer model are frozen, and only the parameters of the adapter are modified. The performance measured in accuracy or F1-score is similar to full fine-tuning on most tasks (Peters et al., 2019). In dialog systems research, adapters were used for the tasks dialog state tracking, response retrieval (Hung et al., 2021) and neural end-to-end dialog (Madotto et al., 2020), but the resource efficiency was not investigated by these works.

2.3 Modular Dialog Systems

Dialog systems are often composed of multiple sub-dialog systems in practical applications. Nehring and Ahmed (2021) presented the modular dialog system (MDS) architecture to combine several dialog systems into one. Each dialog system is called a module and performs its NLU downstream task. The module selection (MS) component selects the appropriate module for each user utterance. MS is a text classification task very similar to ID. The deployment of large NLU models is also an issue in MDS. A MDS with 5 modules uses 6 models ($5 \times$ NLU and $1 \times$ MS). Using our approach, one can reduce the entire MDS's memory requirements considerably.

3 Experiments

3.1 Datasets

We use three datasets in our experiments. We compute the performance metric of the NLU models separately for each dataset.

The HWU64 dataset (Liu et al., 2019) is a dataset for NLU. It contains 25k user utterances from the domain of personal assistants, such as "set my alarm to 07:30" or "play next song". HWU64 spans 64 intents and 21 domains. NLU models from Mehri and Eric (2021) were trained on the DialoGLUE dataset (Mehri et al., 2020) which comprises several datasets, including HWU64. However, the DialoGLUE version of HWU64 contains

only 11k user utterances[5]. For better comparability, we refer to this dataset as HWU64-DialoGLUE and compute the performance metrics of our models for HWU64 and HWU-DialoGLUE.

Another dataset for ID is CLINC150 (Larson et al., 2019). It contains 20k user utterances spanning 150 intents and ten domains. CLINC150 also includes out-of-scope utterances that do not belong to any intents, but are not used in our experiment. The original version and the DialoGLUE version of CLINC150 are identical.

In both datasets, each utterance has exactly one intent label. We split the datasets into train, valid, and test partitions. In CLINC150 and HWU64-DialoGLUE, we reuse the train / valid / test split given by DialoGLUE. In HWU64, we split the data into 80% train, 10% valid and 10% test.

3.2 Models

We conduct our experiments on two models. The BERT model is the standard BERT architecture (Devlin et al., 2019) with a sequence classification head. The output layer consists of one neuron for each target label. We use the implementation of the Hugging Face transformers library (Wolf et al., 2020).

For the BERT+Adapter model, we use the implementation from Pfeiffer et al. (2021).

Further, we compare our models to other models for ID. Mehri et al. (2020) used a BERT model similar to ours for NLU. Further, Mehri et al. (2020) proposed ConvBERT, a BERT model that is pretrained on conversational text instead of on general-purpose text. The best performing model is, to the best of our knowledge, a ConvBERT model with additional *observers* and *examples* (Mehri and Eric, 2021). Observers are tokens that are not attended to and are an alternative to the `[CLS]` token as a semantic representation of utterances *examples* (Mehri and Eric, 2021). Example-driven training improves the prediction step by assigning inputs with similar semantic meanings a similar representation in the semantic space *examples* (Mehri and Eric, 2021).

3.3 Experimental setup

We measure the quality of ID using accuracy, in line with Mehri and Eric (2021); Mehri et al. (2020); Casanueva et al. (2020); Henderson et al. (2020); Bunk et al. (2020), in both the modular and the non-modular scenario. The non-modular scenario is the standard ID setting in which a single DS processes the whole dataset. In the modular scenario, we split the dataset into multiple parts which we use to generate modules in a MDS. We split the dataset along the domains instead of the intents, i.e. all intents of one domain belong to a single module. We generated MDS with three and ten modules.

We use the same model architecture for ID and MS for the modular experiments. MS models are trained on the same training sets as the ID models, but they use the domain labels as target labels. Further, we perform a grid search to find optimal hyperparameters for learning rate and training batch size using the HWU64-DialoGLUE dataset. The grid search determined a learning rate of 10^{-5} for BERT and 10^{-3} for BERT+Adapter and a batch size of 256. We also apply the optimal hyperparameters obtained on this dataset to the experiments on the other datasets. To determine the number of training epochs, we measure the accuracy of ID on the validation set after each training epoch and stop training as soon as the validation accuracy increases by less than 0.2% compared to the last run. In order to measure the models' sizes, we serialized them to disk and measured their size in MB.

We train and evaluate the models and measure the execution time on the HWU64-DialoGLUE dataset. First, we measure the training and inference duration [6]. Then, we calculate how many samples are processed per second (SPS). SPS makes it easier to compare the training duration of differently sized datasets to each other. We repeat this process ten times and average the resulting durations. All experiments were run on a machine with an RTX6000 graphics card.

4 Results

4.1 Model size

While BERT requires 417.75 MB and DIET 63.00 MB for an ID model trained on the HWU64 dataset with 64 intent labels, an adapter only needs 5.87 MB of memory. Figure 1 shows how the frameworks scale in terms of memory when the number of models grows. DIET has the smallest memory requirements when deploying a single model, BERT is considerably higher, and BERT+Adapter

[5] We contacted the author, and he confirmed that this is a bug.

[6] We measured the time for processing samples only, excluding other components of training and inference, such as initially loading a model from disk to memory.

Model	Dataset	SPS train	SPS inference
BERT	CLINC150	8.67 (0.06)	792.17 (5.10)
BERT	HWU64	8.20 (0.06)	777.59 (20.45)
BERT	HWU64-DialoGLUE	16.44 (0.10)	791.26 (6.45)
average BERT		**11.10**	**787.01**
BERT+Adapter	CLINC150	132.35 (0.52)	752.39 (7.15)
BERT+Adapter	HWU64	138.96 (1.21)	750.19 (19.59)
BERT+Adapter	HWU64-DialoGLUE	209.28 (1.21)	762.72 (4.40)
average BERT+Adapter		**160.20**	**755.10**

Table 1: Processing speed of the different models and dataset in samples per second (SPS). The first number is the mean SPS over ten runs, the second number in brackets is the standard deviation over the ten runs. The table lists the average SPS for each model over all datasets.

is slightly higher than BERT. However, when the number of modules grows, the vanilla BERT setup does not scale well. While the slopes of both BERT+Adapter and Rasa are much smaller, BERT+Adapter outperforms Rasa after seven modules.

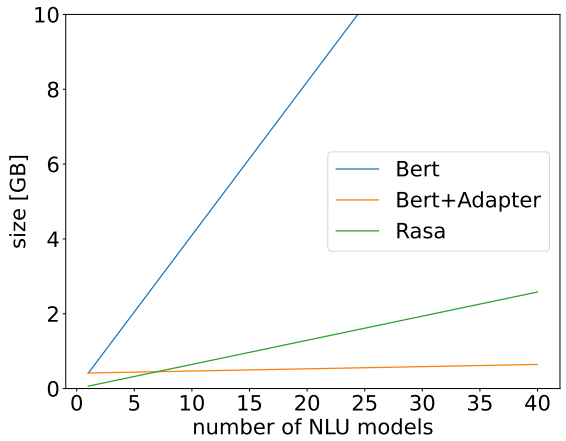

Figure 1: Comparison of the total amounts of memory needed to store the respective framework in GB.

4.2 Training and interference time

Table 1 shows SPS, the number of samples processed per second during training and inference. BERT+Adapter trains considerably faster. BERT+Adapter is slightly faster at inference.

4.3 Quality of intent detection

Table 2 shows the accuracy values of various models for the task of intent detection in both the non-modular scenario (number of modules 1) and the modular scenario (number of modules 3 and 10). Mehri and Eric (2021) achieve the best results across both HWU64-DialoGLUE and CLINC150. We list two results of BERT models, one from our experiments and one from Mehri et al. (2020). We conclude that this difference is due to different hyperparameters used during training.

BERT+Adapter achieves higher scores than BERT in our implementation and similar results to BERT in the implementation of Mehri and Eric (2021). Moreover, in the modular setting, the BERT+Adapter model shows a better performance than the BERT model.

5 Discussion

The model sizes (section 4.1) show that the DIET architecture is most resource-efficient for the deployment of fewer than five models. On the other hand, when the number of models is above six, the BERT+Adapter architecture is more efficient. In addition, the memory saving increases when more models are deployed which is particularly important for chatbot-as-a-service platforms.

SPS improves significantly for training and slightly for inference. We hypothesize that the difference in training time results from the substantially smaller number of parameters that have to be trained in BERT+Adapter, compared to vanilla BERT. Shorter training times are generally favorable. In the chatbot-as-a-service scenario, a shorter training time is helpful for the designer of the DS, because he does not have to wait for training and has a better user experience.

The experiment shows that the accuracy of ID of BERT+Adapter is comparable to the performance of a BERT model. Still, both models do not meet the state of the art. We leave it for future research to find out if the ConvBERT+ model is compatible

Num. Modules	Model	HWU64	HWU64-DialoGLUE	CLINC150
1	BERT	87.65%	85.50%	91.00%
	BERT+Adapter	**89.60%**	89.03%	93.04%
	BERT (Mehri et al., 2020)	-	89.97%	95.93%
	ConvBERT (Mehri et al., 2020)	-	90.43%	97.07%
	ConvBERT+ (Mehri and Eric, 2021)	-	**93.03%**	**97.31%**
3	BERT	84.94%	83.92%	88.80%
	BERT+Adapter	**89.04%**	**86.25%**	**91.00%**
10	BERT	83.07%	80.67%	88.00%
	BERT+Adapter	**88.80%**	**83.55%**	**93.31%**

Table 2: Accuracy values as percentage of different models, datasets and number of modules. Number of modules=1 is the non-modular scenario. The best performing values are highlighted in bold.

with adapters while still performing just as well.

6 Practical considerations on the deployment of NLU models with adapters

Regarding the deployment of NLU models with adapters in a real-world settings, we assume a scenario in which a company offers many user-generated chatbots that are all different from each other in their training data and use case and potentially even come in many languages.

To host 10 languages with 100 chatbots each, we would count $10 \times$ Size BERT $+ 10 \times 100 \times$ Size of Adapter $\approx$ 11GB. 11 GB fit in the memory of a single commercial GPU. The server that hosts the 1000 NLU models would run at all times, keeping the models in memory. In the case of heavy usage, one can horizontally scale out this architecture across several GPUs or several servers.

One downside of our architecture is security and multi-tenant considerations. In multi-tenant environments, it is often a requirement to separate the data of different clients in the architecture, such that data of different clients never get mixed in the same pipeline. The sharing approach suggested in this paper would require a separate pre-trained BERT model for each tenant and language in the multi-tenant environment, decreasing resource savings.

7 Conclusion and Future Work

We examined if the BERT+Adapter architecture is favorable for deploying NLP models compared to BERT and DIET. BERT+Adapter can save a considerable amount of memory compared to BERT and DIET. Also, BERT+Adapter is considerably faster than BERT. The accuracy of ID of BERT+Adapter is still comparable to vanilla BERT.

To the best of our knowledge, our work is the first to point out that adapters can host many NLU models more efficiently. We believe that this approach should be applicable in any environment that hosts multiple pre-trained models. It is especially useful in environments with a skewed usage scheme, meaning that only a few models are used often, and many models are used rarely, such as a chatbots-as-a-service environment. Further, we believe it is applicable in many settings where users can generate and fine-tune their models, e.g. named entity recognition as a service with user-generated content. Finally, adapters are not limited to models with the same task. They can also be used in environments where models for different tasks are deployed (Houlsby et al., 2019), e.g. one adapter for Named Entity Recognition and another adapter for Sentiment Analysis that share the same pre-trained Transformer model.

References

Tanja Bunk, Daksh Varshneya, Vladimir Vlasov, and Alan Nichol. 2020. DIET: Lightweight language understanding for dialogue systems. *arXiv*.

Iñigo Casanueva, Tadas Temčinas, Daniela Gerz, Matthew Henderson, and Ivan Vulić. 2020. Efficient intent detection with dual sentence encoders. In *Proceedings of the 2nd Workshop on Natural Language Processing for Conversational AI*, pages 38–45, Online. Association for Computational Linguistics.

Daniel Cer, Yinfei Yang, Sheng yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St. John, Noah Constant,

Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, Yun-Hsuan Sung, Brian Strope, and Ray Kurzweil. 2018. Universal sentence encoder. *arXiv*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Matthew Henderson, Iñigo Casanueva, Nikola Mrkšić, Pei-Hao Su, Tsung-Hsien Wen, and Ivan Vulić. 2020. ConveRT: Efficient and accurate conversational representations from transformers. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2161–2174, Online. Association for Computational Linguistics.

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. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 2790–2799. PMLR.

Chia-Chien Hung, Anne Lauscher, Simone Paolo Ponzetto, and Goran Glavaš. 2021. DS-TOD: Efficient domain specialization for task oriented dialog. *arXiv*.

Stefan Larson, Anish Mahendran, Joseph J. Peper, Christopher Clarke, Andrew Lee, Parker Hill, Jonathan K. Kummerfeld, Kevin Leach, Michael A. Laurenzano, Lingjia Tang, and Jason Mars. 2019. An evaluation dataset for intent classification and out-of-scope prediction. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1311–1316, Hong Kong, China. Association for Computational Linguistics.

Xingkun Liu, Arash Eshghi, Pawel Swietojanski, and Verena Rieser. 2019. Benchmarking Natural Language Understanding Services for building Conversational Agents. In *Proceedings of the Tenth International Workshop on Spoken Dialogue Systems Technology (IWSDS)*, Ortigia, Siracusa (SR), Italy. Springer.

Andrea Madotto, Zhaojiang Lin, Yejin Bang, and Pascale Fung. 2020. The adapter-bot: All-in-one controllable conversational model. *arXiv*.

Shikib Mehri and Mihail Eric. 2021. Example-driven intent prediction with observers. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2979–2992, Online. Association for Computational Linguistics.

Shikib Mehri, Mihail Eric, and Dilek Hakkani-Tur. 2020. DialoGLUE: A natural language understanding benchmark for task-oriented dialogue. *ArXiv*.

Jan Nehring and Akhyar Ahmed. 2021. Normalisierungsmethoden für Intent Erkennung Modularer Dialogsysteme. In *Tagungsband der 32. Konferenz. Elektronische Sprachsignalverarbeitung (ESSV-2021), March 3-5, Berlin, Germany*. TUDpress.

Matthew E. Peters, Sebastian Ruder, and Noah A. Smith. 2019. To tune or not to tune? adapting pretrained representations to diverse tasks. In *Proceedings of the 4th Workshop on Representation Learning for NLP (RepL4NLP-2019)*, pages 7–14, Florence, Italy. Association for Computational Linguistics.

Jonas Pfeiffer, Aishwarya Kamath, Andreas Rücklé, Kyunghyun Cho, and Iryna Gurevych. 2021. AdapterFusion: Non-destructive task composition for transfer learning. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 487–503, Online. 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. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 46–54, Online. Association for Computational Linguistics.

Sylvestre-Alvise Rebuffi, Hakan Bilen, and Andrea Vedaldi. 2017. Learning multiple visual domains with residual adapters. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Emma Strubell, Ananya Ganesh, and Andrew McCallum. 2019. Energy and policy considerations for deep learning in NLP. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3645–3650, Florence, Italy. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5999–6009. Curran Associates, Inc.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System*

Demonstrations, pages 38–45, Online. Association for Computational Linguistics.