The Microsoft Toolkit of Multi-Task Deep Neural Networks for Natural Language Understanding

Xiaodong Liu[*], **Yu Wang**[*], **Jianshu Ji, Hao Cheng, Xueyun Zhu, Emmanuel Awa,**
Pengcheng He, Weizhu Chen, Hoifung Poon, Guihong Cao and Jianfeng Gao
Microsoft Corporation
{xiaodl,yuwan,jianshuj,chehao,xuzhu}@microsoft.com

Abstract

We present MT-DNN[1], an open-source natural language understanding (NLU) toolkit that makes it easy for researchers and developers to train customized deep learning models. Built upon PyTorch and Transformers, MT-DNN is designed to facilitate rapid customization for a broad spectrum of NLU tasks, using a variety of objectives (classification, regression, structured prediction) and text encoders (e.g., RNNs, BERT, RoBERTa, UniLM). A unique feature of MT-DNN is its built-in support for robust and transferable learning using the adversarial multi-task learning paradigm. To enable efficient production deployment, MT-DNN supports multi-task knowledge distillation, which can substantially compress a deep neural model without significant performance drop. We demonstrate the effectiveness of MT-DNN on a wide range of NLU applications across general and biomedical domains. The software and pre-trained models will be publicly available at https://github.com/namisan/mt-dnn.

1 Introduction

NLP model development has observed a paradigm shift in recent years, due to the success in using pre-trained language models to improve a wide range of NLP tasks (Peters et al., 2018; Devlin et al., 2019). Unlike the traditional pipeline approach that conducts annotation in stages using primarily supervised learning, the new paradigm features a universal **pretraining** stage that trains a large neural language model via self-supervision on a large unlabeled text corpus, followed by a **fine-tuning** step that starts from the pretrained contextual representations and conducts supervised learning for individual tasks. The pretrained language models can effectively model textual variations and distributional similarity. Therefore, they can make subsequent task-specific training more sample efficient and often significantly boost performance in downstream tasks. However, these models are quite large and pose significant challenges to production deployment that has stringent memory or speed requirements. As a result, **knowledge distillation** has become another key feature in this new learning paradigm. An effective distillation step can often substantially compress a large model for efficient deployment (Clark et al., 2019; Tang et al., 2019; Liu et al., 2019a).

In the NLP community, there are several well designed frameworks for research and commercial purposes, including toolkits for providing conventional layered linguistic annotations (Manning et al., 2014), platforms for developing novel neural models (Gardner et al., 2018) and systems for neural machine translation (Ott et al., 2019). However, it is hard to find an existing tool that supports all features in the new paradigm and can be easily customized for new tasks. For example, (Wolf et al., 2019) provides a number of popular Transformer-based (Vaswani et al., 2017) text encoders in a nice unified interface, but does not offer multi-task learning or adversarial training, state-of-the-art techniques that have been shown to significantly improve performance. Additionally, most public frameworks do not offer knowledge distillation. A notable exception is DistillBERT (Sanh et al., 2019), but it provides a standalone compressed model and does not support task-specific model compression that can further improve performance.

We introduce MT-DNN, a comprehensive and easily-configurable open-source toolkit for building robust and transferable natural language understanding models. MT-DNN is built upon PyTorch (Paszke et al., 2019) and the popular Transformer-

---

[*]Equal Contribution.

[1]The complete name of our toolkit is MT^2-*DNN* (The **M**icrosoft **T**oolkit of **M**ulti-**T**ask **D**eep **N**eural **N**etworks for Natural Language Understanding), but we use MT-DNN for sake of simplicity.

based text-encoder interface (Wolf et al., 2019). It supports a large inventory of pretrained models, neural architectures, and NLU tasks, and can be easily customized for new tasks.

A key distinct feature for MT-DNN is that it provides out-of-box adversarial training, multi-task learning, and knowledge distillation. Users can train a set of related tasks jointly to amplify each other. They can also invoke adversarial training (Miyato et al., 2018; Jiang et al., 2019; Liu et al., 2020), which helps improve model robustness and generalizability. For production deployment where large model size becomes a practical obstacle, users can use MT-DNN to compress the original models into substantially smaller ones, even using a completely different architecture (e.g., compressed BERT or other Transformer-based text encoders into LSTMs (Hochreiter and Schmidhuber, 1997)). The distillation step can similarly leverage multi-task learning and adversarial training. Users can also conduct pretraining from scratch using the masked language model objective in MT-DNN. Moreover, in the fine-tuning step, users can incorporate this as an auxiliary task on the training text, which has been shown to improve performance. MT-DNN provides a comprehensive list of state-of-the-art pre-trained NLU models, together with step-by-step tutorials for using such models in general and biomedical applications.

2 Design

MT-DNN is designed for modularity, flexibility, and ease of use. These modules are built upon PyTorch (Paszke et al., 2019) and Transformers (Wolf et al., 2019), allowing the use of the SOTA pre-trained models, e.g., BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019c) and UniLM (Dong et al., 2019). The unique attribute of this package is a flexible interface for adversarial multi-task fine-tuning and knowledge distillation, so that researchers and developers can build large SOTA NLU models and then compress them to small ones for online deployment.The overall workflow and system architecture are shown in Figure 1 and Figure 3 respectively.

2.1 Workflow

As shown in Figure 1, starting from the neural language model pre-training, there are three different training configurations by following the directed arrows:
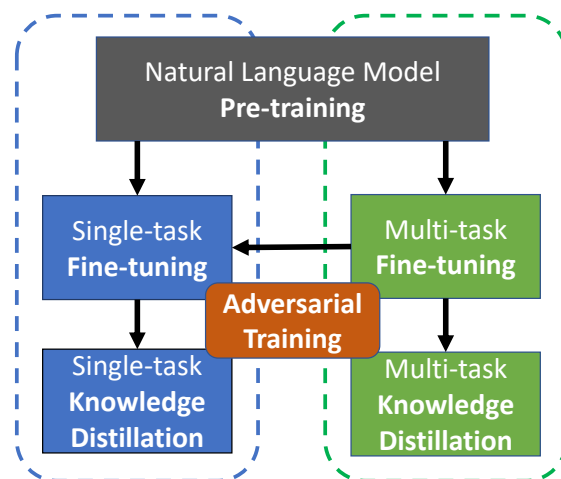
- Single-task configuration: single-task fine-

Figure 1: The workflow of MT-DNN: train a neural language model on a large amount of unlabeled raw text to obtain general contextual representations; then fine-tune the learned contextual representation on downstream tasks, e.g. GLUE (Wang et al., 2018); lastly, distill this large model to a lighter one for online deployment. In the later two phrases, we can leverage powerful multi-task learning and adversarial training to further improve performance.

tuning and single-task knowledge distillation;

- Multi-task configuration: multi-task fine-tuning and multi-task knowledge distillation;

- Multi-stage configuration: multi-task fine-tuning, single-task fine tuning and single-task knowledge distillation.

Moreover, all configurations can be additionally equipped with the adversarial training. Each stage of the workflow is described in details as follows.

Neural Language Model Pre-Training Due to the great success of deep contextual representations, such as ELMo (Peters et al., 2018), GPT (Radford et al., 2018) and BERT (Devlin et al., 2019), it is common practice of developing NLU models by first pre-training the underlying neural text representations (text encoders) through massive language modeling which results in superior text representations transferable across multiple NLP tasks. Because of this, there has been an increasing effort to develop better pre-trained text encoders by multiplying either the scale of data (Liu et al., 2019c) or the size of model (Raffel et al., 2019). Similar to existing codebases (Devlin et al., 2019), MT-DNN supports the LM pre-training from scratch with multiple types of objectives, such as masked LM (Devlin et al., 2019) and
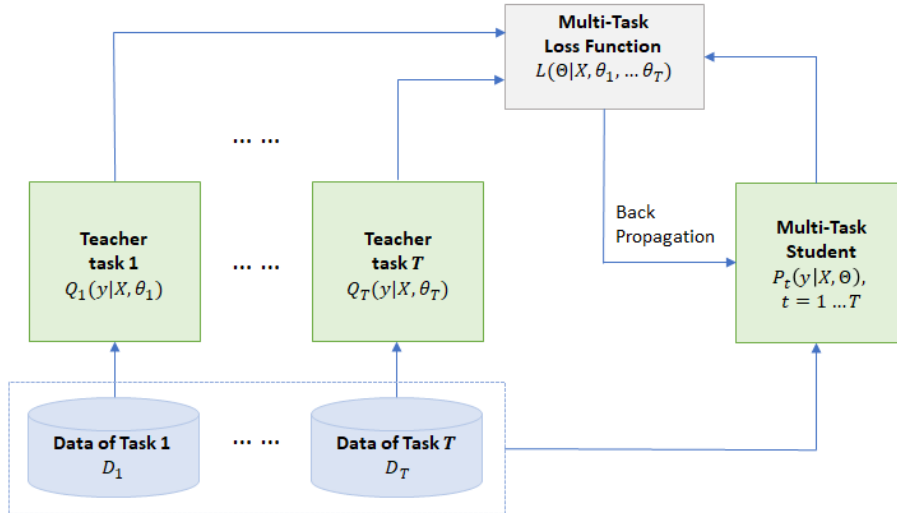
Figure 2: Process of knowledge distillation for MTL. A set of tasks where there is task-specific labeled training data are picked. Then, for each task, an ensemble of different neural nets (teacher) is trained. The teacher is used to generate for each task-specific training sample a set of soft targets. Given the soft targets of the training datasets across multiple tasks, a single MT-DNN (student) shown in Figure 3 is trained using multi-task learning and back propagation, except that if task t has a teacher, the task-specific loss is the average of two objective functions, one for the correct targets and the other for the soft targets assigned by the teacher.

next sentence prediction (Devlin et al., 2019).

Moreover, users can leverage the LM pre-training, such as masked LM used by BERT, as an auxiliary task for fine-tuning under the multi-task learning (MTL) framework (Sun et al., 2019; Liu et al., 2019b).

Fine-tuning Once the text encoder is trained in the pre-training stage, an additional task-specific layer is usually added for fine-tuning based on the down-stream task. Besides the existing typical single-task fine-tuning, MT-DNN facilitates a joint fine-tuning with a configurable list of related tasks in a MTL fashion. By encoding task-relatedness and sharing underlying text representations, MTL is a powerful training paradigm that promotes the model generalization ability and results in improved performance (Caruana, 1997; Liu et al., 2019b; Luong et al., 2015; Liu et al., 2015; Ruder, 2017; Collobert et al., 2011). Additionally, a two-step fine-tuning stage is also supported to utilize datasets from related tasks, i.e. a single-task fine-tuning following a multi-task fine-tuning. It also supports two popular sampling strategies in MTL training: 1) sampling tasks uniformly (Caruana, 1997; Liu et al., 2015); 2) sampling tasks based on the size of the dataset (Liu et al., 2019b). This makes it easy to explore various ways to feed training data to MTL training. Finally, to further improve the model robustness, MT-DNN also offers a recipe to apply adversarial training (Madry et al., 2017; Zhu et al., 2019; Jiang et al., 2019) in the fine-tuning stage.

Knowledge Distillation Although contextual text representation models pre-trained with massive text data have led to remarkable progress in NLP, it is computationally prohibitive and inefficient to deploy such models with millions of parameters for real-world applications (e.g. BERT large model has 344 million parameters). Therefore, in order to expedite the NLU model learned in either a single-task or multi-task fashion for deployment, MT-DNN additionally supports the multi-task knowledge distillation (Clark et al., 2019; Liu et al., 2019a; Tang et al., 2019; Balan et al., 2015; Ba and Caruana, 2014), an extension of (Hinton et al., 2015), to compress cumbersome models into lighter ones. The multi-task knowledge distillation process is illustrated in Figure 2. Similar to the fine-tuning stage, adversarial training is available in the knowledge distillation stage.

2.2 Architecture

Lexicon Encoder (l_1)**:** The input $X = \{x_1, ..., x_m\}$ is a sequence of tokens of length m. The first token x_1 is always a specific token, e.g. [CLS] for BERT Devlin et al. (2019) while <s> for RoBERTa Liu et al. (2019c). If X is a pair of sentences (X_1, X_2), we separate these sentences with special tokens, e.g. [SEP] for BERT and [</s>] for RoBERTa. The lexicon encoder maps X into a sequence of input embedding vectors,
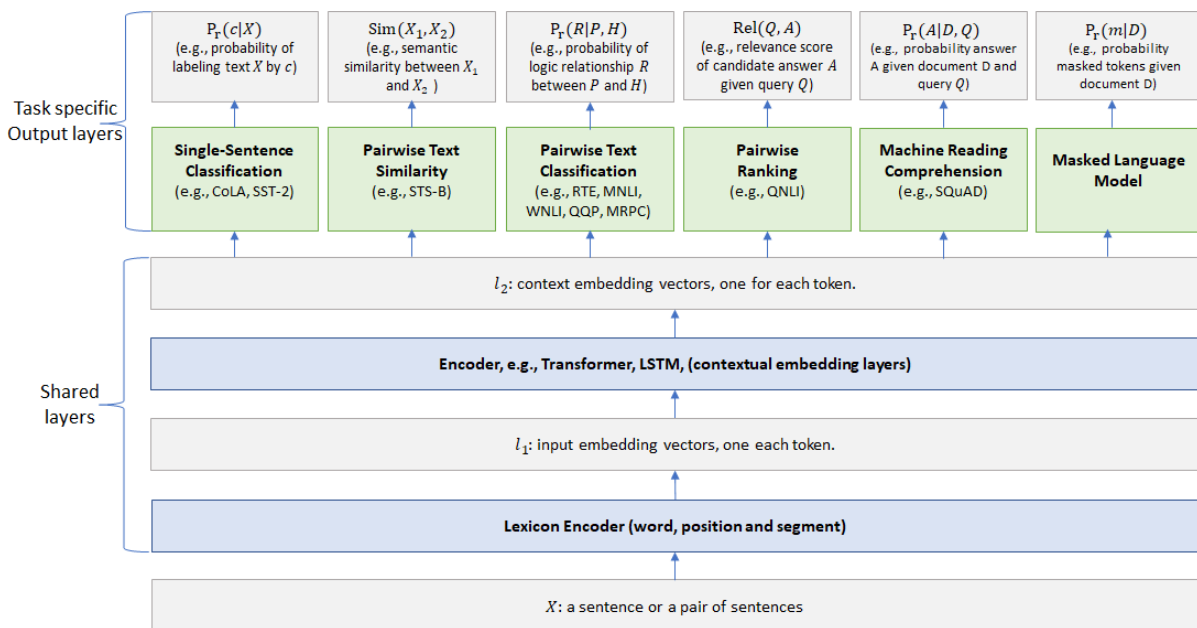
120

Figure 3: Overall System Architecture: The lower layers are shared across all tasks while the top layers are task-specific. The input X (either a sentence or a set of sentences) is first represented as a sequence of embedding vectors, one for each word, in l_1. Then the encoder, e.g a Transformer or recurrent neural network (LSTM) model, captures the contextual information for each word and generates the shared contextual embedding vectors in l_2. Finally, for each task, additional task-specific layers generate task-specific representations, followed by operations necessary for classification, similarity scoring, or relevance ranking. In case of adversarial training, we perturb embeddings from the lexicon encoder and then add an extra loss term during the training. Note that for the inference phrase, it does not require perturbations.

one for each token, constructed by summing the corresponding word with positional, and optional segment embeddings.

Encoder (l_2): We support a multi-layer bidirectional Transformer (Vaswani et al., 2017) or a LSTM (Hochreiter and Schmidhuber, 1997) encoder to map the input representation vectors (l_1) into a sequence of contextual embedding vectors $\mathbf{C} \in R^{d \times m}$. This is the shared representation across different tasks. Note that MT-DNN also allows developers to customize their own encoders. For example, one can design an encoder with few Transformer layers (e.g. 3 layers) to distill knowledge from the BERT large model (24 layers), so that they can deploy this small mode online to meet the latency restriction as shown in Figure 2.

Task-Specific Output Layers: We can incorporate arbitrary natural language tasks, each with its task-specific output layer. For example, we implement the output layers as a neural decoder for a neural ranker for relevance ranking, a logistic regression for text classification, and so on. A multi-step reasoning decoder, SAN (Liu et al., 2018a,b) is also provided. Customers can choose from existing task-specific output layer or implement new

one by themselves.

3 Application

In this section, we present a comprehensive set of examples to illustrate how to customize MT-DNN for new tasks. We use popular benchmarks from general and biomedical domains, including GLUE (Wang et al., 2018), SNLI (Bowman et al., 2015), SciTail (Khot et al., 2018), SQuAD (Rajpurkar et al., 2016), ANLI (Nie et al., 2019), and biomedical named entity recognition (NER), relation extraction (RE) and question answering (QA) (Lee et al., 2019). To make the experiments reproducible, we make all the configuration files publicly available. We also provide a quick guide for customizing a new task in Jupyter notebooks.

3.1 General Domain Natural Language Understanding Benchmarks

• **GLUE**. The General Language Understanding Evaluation (GLUE) benchmark is a collection of nine natural language understanding (NLU) tasks. As shown in Table 1, it includes question answering (Rajpurkar et al., 2016), linguistic acceptability (Warstadt et al., 2018), sentiment analy-

Corpus	Task	Formulation
Corpus	Task	Formulation
GLUE		
CoLA	Acceptability	Classification
SST	Sentiment	Classification
MNLI	NLI	Classification
RTE	NLI	Classification
WNLI	NLI	Classification
QQP	Paraphrase	Classification
MRPC	Paraphrase	Classification
QNLI	QA/NLI	Classification
QNLI v1.0	QA/NLI	Pairwise Ranking
STS-B	Similarity	Regression
Others		
SNLI	NLI	Classification
SciTail	NLI	Classification
ANLI	NLI	Classification
SQuAD	MRC	Span Classification

Table 1: Summary of the four benchmarks: GLUE, SNLI, SciTail and ANLI.

Model	MNLI Acc	RTE Acc	QNLI Acc	SST Acc	MRPC F1
BERT	84.5	63.5	91.1	92.9	89.0
BERT + MTL	85.3	79.1	91.5	93.6	89.2
BERT + AdvTrain	85.6	71.2	91.6	93.0	91.3

Table 2: Comparison among single task, multi-Task and adversarial training on MNLI, RTE, QNLI, SST and MPRC in GLUE.

Model	Dev	Test
BERT$_{LARGE}$ (Nie et al., 2019)	49.3	44.2
RoBERTa$_{LARGE}$ (Nie et al., 2019)	53.7	49.7
RoBERTa-LARGE + AdvTrain	57.1	57.1

Table 3: Results in terms of accuracy on the ANLI.

sis (Socher et al., 2013), text similarity (Cer et al., 2017), paraphrase detection (Dolan and Brockett, 2005), and natural language inference (NLI) (Dagan et al., 2006; Bar-Haim et al., 2006; Giampiccolo et al., 2007; Bentivogli et al., 2009; Levesque et al., 2012; Williams et al., 2018). The diversity of the tasks makes GLUE very suitable for evaluating the generalization and robustness of NLU models.

• **SNLI**. The Stanford Natural Language Inference (SNLI) dataset contains 570k human annotated sentence pairs, in which the premises are drawn from the captions of the Flickr30 corpus and hypothe-

ses are manually annotated (Bowman et al., 2015). This is the most widely used entailment dataset for NLI.

• **SciTail** This is a textual entailment dataset derived from a science question answering (SciQ) dataset (Khot et al., 2018). In contrast to other entailment datasets mentioned previously, the hypotheses in SciTail are created from science questions while the corresponding answer candidates and premises come from relevant web sentences retrieved from a large corpus.

• **ANLI**. The Adversarial Natural Language Inference (ANLI, Nie et al. (2019)) is a new large-scale NLI benchmark dataset, collected via an iterative, adversarial human-and-model-in-the-loop procedure. Particular, the data is selected to be difficult to the state-of-the-art models, including BERT and RoBERTa.

• **SQuAD**. The Stanford Question Answering Dataset (SQuAD) (Rajpurkar et al., 2016) contains about 23K passages and 100K questions. The passages come from approximately 500 Wikipedia articles and the questions and answers are obtained by crowdsourcing.

Following (Devlin et al., 2019), table 2 compares different training algorithm: 1) BERT denotes a single task fine-tuning; 2) BERT + MTL indicates that it is trained jointly via MTL; at last 3), BERT + AdvTrain represents that a single task fine-tuning with adversarial training. It is obvious that the both MLT and adversarial training helps to obtain a better result. We further test our model on an adversarial natural language inference (ANLI) dataset (Nie et al., 2019). Table 3 summarizes the results on ANLI. As Nie et al. (2019), all the dataset of ANLI (Nie et al., 2019), MNLI (Williams et al., 2018), SNLI (Bowman et al., 2015) and FEVER (Thorne et al., 2018) are combined as training. RoBERTa-LARGE+AdvTrain obtains the best performance compared with all the strong baselines, demonstrating the advantage of adversarial training.

3.2 Biomedical Natural Language Understating Benchmarks

There has been rising interest in exploring natural language understanding tasks in high-value domains other than newswire and the Web. In our release, we provide MT-DNN customization for three representative biomedical natural language understanding tasks:

• Named entity recognition (NER): In biomedical natural language understanding, NER has received

greater attention than other tasks and datasets are available for recognizing various biomedical entities such as disease, gene, drug (chemical).

• Relation extraction (RE): Relation extraction is more closely related to end applications, but annotation effort is significantly higher compared to NER. Most existing RE tasks focus on binary relations within a short text span such as a sentence of an abstract. Examples include gene-disease or protein-chemical relations.

• Question answering (QA): Inspired by interest in QA for the general domain, there has been some effort to create question-answering datasets in biomedicine. Annotation requires domain expertise, so it is significantly harder than in general domain, where it is to produce large-scale datasets by crowdsourcing.

The MT-DNN customization can work with standard or biomedicine-specific pretraining models such as BioBERT, and can be directly applied to biomedical benchmarks (Lee et al., 2019).

3.3 Extension

```
 1  snli:
 2    data_format: PremiseAndOneHypothesis
 3    task_layer_type: LinearLayer
 4    labels:
 5    - contradiction
 6    - neutral
 7    - entailment
 8    metric_meta:
 9    - ACC
10    loss: CeCriterion
11    kd_loss: MseCriterion
12    adv_loss: KlCriterion
13    n_class: 3
14    task_type: Classification
```

Figure 4: The configuration of SNLI.

We will go though a typical Natural Language Inference task, e.g. SNLI, which is one of the most popular benchmark, showing how to apply our toolkit to a new task. MT-DNN is driven by configuration and command line arguments. Firstly, the SNLI configuration is shown in Figure 4. The configuration defines tasks, model architecture as well as loss functions. We briefly introduce these attributes as follows:

1. *data_format* is a required attribute and it denotes that each sample includes two sentences (premise and hypothesis). Please refer the tutorial and API for supported formats.

2. *task_layer_type* specifies architecture of the task specific layer. The default is a "linear layer".

3. *labels* Users can list unique values of labels. The configuration helps to convert back and forth between text labels and numbers during training and evaluation. Without it, MT-DNN assumes the label of prediction are numbers.

4. *metric_meta* is the evaluation metric used for validation.

5. *loss* is the loss function for SNLI. It also supports other functions, e.g. MSE for regression.

6. *kd_loss* is the loss function in the knowledge distillation setting.

7. *adv_loss* is the loss function in the adversarial setting.

8. *n_class* denotes the number of categories for SNLI.

9. *task_type* specifies whether it is a classification task or a regression task.

Once the configuration is provided, one can train the customized model for the task, using any supported pre-trained models as starting point.

MT-DNN is also highly extensible, as shown in Figure 4, *loss* and *task_layer_type* point to existing classes in code. Users can write customized classes and plug into MT-DNN. The customized classes could then be used via configuration.

4 Conclusion

Microsoft MT-DNN is an open-source natural language understanding toolkit which facilitates researchers and developers to build customized deep learning models. Its key features are: 1) support for robust and transferable learning using adversarial multi-task learning paradigm; 2) enable knowledge distillation under the multi-task learning setting which can be leveraged to derive lighter models for efficient online deployment. We will extend MT-DNN to support Natural Language Generation tasks, e.g. Question Generation, and incorporate more pre-trained encoders, e.g. T5 (Raffel et al., 2019) in future.

References

Jimmy Ba and Rich Caruana. 2014. Do deep nets really need to be deep? In *Advances in neural information processing systems*, pages 2654–2662.

Anoop Korattikara Balan, Vivek Rathod, Kevin P Murphy, and Max Welling. 2015. Bayesian dark knowledge. In *Advances in Neural Information Processing Systems*, pages 3438–3446.

Roy Bar-Haim, Ido Dagan, Bill Dolan, Lisa Ferro, and Danilo Giampiccolo. 2006. The second PASCAL recognising textual entailment challenge. In *Proceedings of the Second PASCAL Challenges Workshop on Recognising Textual Entailment*.

Luisa Bentivogli, Ido Dagan, Hoa Trang Dang, Danilo Giampiccolo, and Bernardo Magnini. 2009. The fifth pascal recognizing textual entailment challenge. In *In Proc Text Analysis Conference (TAC'09*.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics.

Rich Caruana. 1997. Multitask learning. *Machine learning*, 28(1):41–75.

Daniel Cer, Mona Diab, Eneko Agirre, Inigo Lopez-Gazpio, and Lucia Specia. 2017. Semeval-2017 task 1: Semantic textual similarity-multilingual and cross-lingual focused evaluation. *arXiv preprint arXiv:1708.00055*.

Kevin Clark, Minh-Thang Luong, Urvashi Khandelwal, Christopher D Manning, and Quoc V Le. 2019. Bam! born-again multi-task networks for natural language understanding. *arXiv preprint arXiv:1907.04829*.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *Journal of machine learning research*, 12(Aug):2493–2537.

Ido Dagan, Oren Glickman, and Bernardo Magnini. 2006. The pascal recognising textual entailment challenge. In *Proceedings of the First International Conference on Machine Learning Challenges: Evaluating Predictive Uncertainty Visual Object Classification, and Recognizing Textual Entailment*, MLCW'05, pages 177–190, Berlin, Heidelberg. Springer-Verlag.

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.

William B Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*.

Li Dong, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. 2019. Unified language model pre-training for natural language understanding and generation. In *Advances in Neural Information Processing Systems*, pages 13042–13054.

Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson Liu, Matthew Peters, Michael Schmitz, and Luke Zettlemoyer. 2018. Allennlp: A deep semantic natural language processing platform. *arXiv preprint arXiv:1803.07640*.

Danilo Giampiccolo, Bernardo Magnini, Ido Dagan, and Bill Dolan. 2007. The third PASCAL recognizing textual entailment challenge. In *Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing*, pages 1–9, Prague. Association for Computational Linguistics.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Tuo Zhao. 2019. Smart: Robust and efficient fine-tuning for pre-trained natural language models through principled regularized optimization. *arXiv preprint arXiv:1911.03437*.

Tushar Khot, Ashish Sabharwal, and Peter Clark. 2018. SciTail: A textual entailment dataset from science question answering. In *AAAI*.

Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. 2019. Biobert: pre-trained biomedical language representation model for biomedical text mining. *arXiv preprint arXiv:1901.08746*.

Hector Levesque, Ernest Davis, and Leora Morgenstern. 2012. The winograd schema challenge. In *Thirteenth International Conference on the Principles of Knowledge Representation and Reasoning*.

Xiaodong Liu, Hao Cheng, Pengcheng He, Weizhu Chen, Yu Wang, Hoifung Poon, and Jianfeng Gao. 2020. Adversarial training for large neural language models. *arXiv preprint arXiv:2004.08994*.

Xiaodong Liu, Kevin Duh, and Jianfeng Gao. 2018a. Stochastic answer networks for natural language inference. *arXiv preprint arXiv:1804.07888*.

Xiaodong Liu, Jianfeng Gao, Xiaodong He, Li Deng, Kevin Duh, and Ye-Yi Wang. 2015. Representation learning using multi-task deep neural networks for semantic classification and information retrieval. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 912–921.

Xiaodong Liu, Pengcheng He, Weizhu Chen, and Jianfeng Gao. 2019a. Improving multi-task deep neural networks via knowledge distillation for natural language understanding. *arXiv preprint arXiv:1904.09482*.

Xiaodong Liu, Pengcheng He, Weizhu Chen, and Jianfeng Gao. 2019b. Multi-task deep neural networks for natural language understanding. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4487–4496, Florence, Italy. Association for Computational Linguistics.

Xiaodong Liu, Yelong Shen, Kevin Duh, and Jianfeng Gao. 2018b. Stochastic answer networks for machine reading comprehension. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019c. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Minh-Thang Luong, Quoc V Le, Ilya Sutskever, Oriol Vinyals, and Lukasz Kaiser. 2015. Multi-task sequence to sequence learning. *arXiv preprint arXiv:1511.06114*.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. 2017. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*.

Christopher D Manning, Mihai Surdeanu, John Bauer, Jenny Rose Finkel, Steven Bethard, and David McClosky. 2014. The stanford corenlp natural language processing toolkit. In *Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations*, pages 55–60.

Takeru Miyato, Shin-ichi Maeda, Masanori Koyama, and Shin Ishii. 2018. Virtual adversarial training: a regularization method for supervised and semi-supervised learning. *IEEE transactions on pattern analysis and machine intelligence*, 41(8):1979–1993.

Yixin Nie, Adina Williams, Emily Dinan, Mohit Bansal, Jason Weston, and Douwe Kiela. 2019. Adversarial nli: A new benchmark for natural language understanding. *arXiv preprint arXiv:1910.14599*.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. *arXiv preprint arXiv:1904.01038*.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, pages 8024–8035.

Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2018. Language models are unsupervised multitask learners.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Sebastian Ruder. 2017. An overview of multi-task learning in deep neural networks. *arXiv preprint arXiv:1706.05098*.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

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.

Yu Sun, Shuohuan Wang, Yukun Li, Shikun Feng, Hao Tian, Hua Wu, and Haifeng Wang. 2019. Ernie 2.0: A continual pre-training framework for language understanding. *arXiv preprint arXiv:1907.12412*.

Raphael Tang, Yao Lu, Linqing Liu, Lili Mou, Olga Vechtomova, and Jimmy Lin. 2019. Distilling task-specific knowledge from bert into simple neural networks. *arXiv preprint arXiv:1903.12136*.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. Fever: a large-scale dataset for fact extraction and verification. *arXiv preprint arXiv:1803.05355*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*.

Alex Warstadt, Amanpreet Singh, and Samuel R Bowman. 2018. Neural network acceptability judgments. *arXiv preprint arXiv:1805.12471*.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. 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 1112–1122. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, R'emi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771.

Chen Zhu, Yu Cheng, Zhe Gan, Siqi Sun, Thomas Goldstein, and Jingjing Liu. 2019. Freelb: Enhanced adversarial training for language understanding. *arXiv preprint arXiv:1909.11764*.