Accelerating Natural Language Understanding in Task-Oriented Dialog

Ojas Ahuja and **Shrey Desai**
Department of Computer Science
The University of Texas at Austin
{ojas, shreydesai}@utexas.edu

Abstract

Task-oriented dialog models typically leverage complex neural architectures and large-scale, pre-trained Transformers to achieve state-of-the-art performance on popular natural language understanding benchmarks. However, these models frequently have in excess of tens of millions of parameters, making them impossible to deploy on-device where resource-efficiency is a major concern. In this work, we show that a simple convolutional model compressed with structured pruning achieves largely comparable results to BERT (Devlin et al., 2019) on ATIS and Snips, with under 100K parameters. Moreover, we perform acceleration experiments on CPUs, where we observe our multi-task model predicts intents and slots nearly 63× faster than even DistilBERT (Sanh et al., 2019).

1 Introduction

The advent of smart devices like Amazon Alexa, Facebook Portal, and Google Assistant has increased the necessity of resource-efficient task-oriented systems (Coucke et al., 2018; Zhang et al., 2020; Desai et al., 2020). These systems chiefly perform two natural language understanding tasks, intent detection and slot filling, where the goals are to understand what the user is trying to accomplish and the metadata associated with the request, respectively (Gupta et al., 2018). However, there remains a disconnect between state-of-the-art task-oriented systems and their deployment in real-world applications. Recent top performing systems have largely saturated performance on ATIS (Hemphill et al., 1990) and Snips (Coucke et al., 2018) by leveraging complex neural architectures and large-scale, pre-trained language models (Devlin et al., 2019), but their usability in on-device settings remains suspect (Qin et al., 2019; Cheng et al., 2017). Mobile phones, for example, have sharp hardware constraints and limited memory capacities, implying systems must optimize for both accuracy and resource-efficiency as possible to be able to run in these types of environments (Lin et al., 2010; McIntosh et al., 2018).

In this work, we present a vastly simplified, single-layer convolutional model (Kim, 2014; Bai et al., 2018) that is highly compressible but nonetheless achieves competitive results on task-oriented natural language understanding benchmarks. In order to compress the model, we use structured magnitude-based pruning (Anwar et al., 2017; Li et al., 2017), a two-step approach where (1) entire convolutional filters are deleted according to their ℓ_2 norms; and (2) remaining portions of the underlying weight matrix are spliced together. The successive reduction in the number of convolutional output connections permits downstream weight matrices to reduce their number of input connections as well, collectively resulting in a smaller model. Structured pruning and re-training steps are then interleaved to ensure the model is able to reconstruct lost filters that may contribute valuable information. During test-time, however, we use the pruned model as-is without further fine-tuning.

Our simple convolutional model with structured pruning obtains strong results despite having less than 100K parameters. On ATIS, our multi-task model achieves 95% intent accuracy and 94% slot F1, only about 2% lower than BERT (Devlin et al., 2019). Structured pruning also admits significantly faster inference: on CPUs, we show our model is 63× faster than DistilBERT. Unlike compression methods based on unstructured pruning (Frankle and Carbin, 2019), our model enjoys speedups *without* having to rely on a sparse tensor library at test-time (Han et al., 2016), thus we demonstrate the potential for usage in resource-constrained, on-device settings. Our code is publicly available at https://github.com/oja/pruned-nlu.

Proceedings of the 2nd Workshop on Natural Language Processing for Conversational AI, pages 46–53
July 9, 2020.

2 Related Work

Task-Oriented Dialog. Dialog systems perform a range of tasks, including language understanding, dialog state tracking, content planning, and text generation (Bobrow et al., 1977; Henderson, 2015; Yu et al., 2016; Yan et al., 2017; Gao et al., 2018). For smart devices, specifically, intent detection and slot filling form the backbone of natural language understanding (NLU) modules, which can either be used in single-turn or multi-turn conversations (Coucke et al., 2018; Rastogi et al., 2020). We contribute a single-turn, multi-task NLU system especially tailored for on-device settings, as demonstrated through acceleration experiments.

Model Compression. In natural language processing, numerous works have used compression techniques like quantization (Wróbel et al., 2018; Zafrir et al., 2019), distillation (Sanh et al., 2019; Tang et al., 2019; Jiao et al., 2020), pruning (Yoon et al., 2018; Gordon et al., 2020), and smaller representations (Ravi and Kozareva, 2018; Kozareva and Ravi, 2018; Desai et al., 2020). Concurrently, Desai et al. (2020) develop lightweight convolutional representations for on-device task-oriented systems, related to our goals, but they do not compare with other compression methods and solely evaluate on a proprietary dataset. In contrast, we compare the efficacy of structured pruning against strong baselines—including BERT (Devlin et al., 2019)—on the open-source ATIS and Snips datasets.

3 Convolutional Model

Convolutions for On-Device Modeling. State-of-the-art task-oriented models are largely based on recurrent neural networks (RNNs) (Wang et al., 2018) or Transformers (Qin et al., 2019). However, these models are often impractical to deploy in low-resource settings. Recurrent models must sequentially unroll sequences during inference, and self-attention mechanisms in Transformers process sequences with quadratic complexity (Vaswani et al., 2017). High-performing, pre-trained Transformers, in particular, also have upwards of tens of millions of parameters, even when distilled (Tang et al., 2019; Sanh et al., 2019).

Convolutional neural networks (CNNs), in contrast, are highly parallelizable and can be significantly compressed with structured pruning (Li et al., 2017), while still achieving competitive performance on a variety of NLP tasks (Kim, 2014; Gehring et al., 2017). Furthermore, the core convolution operation has enjoyed speedups with dedicated digital signal processors (DSPs) and field programmable gate arrays (FPGAs) (Ahmad and Pasha, 2020). Model compatibility with on-device hardware is one of the most important considerations for practitioners as, even if a model works well on high throughput GPUs, its components may saturate valuable resources like memory and power (Lin et al., 2010).

Model Description. Model inputs are encoded as a sequence of integers $\mathbf{w} = (w_1, \cdots, w_n)$ and right-padded up to a maximum sequence length. The embedding layer replaces each token w_i with a corresponding d-dimensional vector $\mathbf{e}_i \in \mathbb{R}^d$ sourced from pre-trained GloVe embeddings (Pennington et al., 2014). A feature map $\mathbf{c} \in \mathbb{R}^{n-h+1}$ is then calculated by applying a convolutional filter of height h over the embedded input sequence. We apply max-over-time pooling $\hat{c} = \max(\mathbf{c})$ (Collobert et al., 2011) to simultaneously reduce the dimensionality and extract the most salient features. The pooled features are then concatenated and fed through a linear layer with dropout (Srivastava et al., 2014). The objective is to maximize the log likelihood of intents, slots, or both (under a multi-task setup), and is optimized with Adam (Kingma and Ba, 2015).

To ensure broad applicability, our model emphasizes simplicity, and therefore minimizes the number of extraneous architectural decisions: there is only a single convolutional block, no residual connections, and no normalization layers.

Temporal Padding. The model described above is capable of predicting an intent that encompasses the entire input sequence, but cannot be used for sequence labeling tasks, namely slot filling. To create a one-to-one correspondence between input tokens and output slots, Bai et al. (2018) left-pad the input sequence by $k-1$, where k is the kernel size. We modify this by loosening the causality constraint and instead padding each side by $\frac{k-1}{2}$. Visually, this results in a "centered" convolution that inculcates bidirectional context when computing a feature map. Note that this padding is unnecessary for intent detection, therefore we skip it when training a single-task intent model.

Multi-Task Training. Intent detection and slot filling can either be disjointly learned with dedicated single-task models or jointly learned with a

unified multi-task model (Liu and Lane, 2016). In the latter model, we introduce task-specific heads on top of the common representation layer and simultaneously optimize both objectives:

$$\mathcal{L}_{\text{joint}} = \alpha \mathcal{L}_{\text{intent}} + (1 - \alpha)\mathcal{L}_{\text{slot}}$$

for α where $0 \leq \alpha \leq 1$. Empirically, we observe that weighting $\mathcal{L}_{\text{slot}}$ more than $\mathcal{L}_{\text{intent}}$ results in higher performance ($\alpha \approx 0.2$). Our hypothesis is that, because of the comparative difficulty of the slot filling task, the model is required to learn a more robust representation of each utterance, which is nonetheless useful for intent detection.

4 Structured Pruning

Structured vs. Unstructured Pruning. Pruning is one compression technique that removes weights from an over-parameterized model (LeCun et al., 1990), often relying on a heuristic function that ranks weights (or groups of weights) by their importance. Methods for pruning are broadly categorized as unstructured and structured: unstructured pruning allows weights to be removed haphazardly without geometric constraints, but structured pruning induces well-defined sparsity patterns, for example, dropping entire filters in a convolutional layer according to their norm (Molchanov et al., 2016; Li et al., 2017; Anwar et al., 2017). Critically, **the model's true size is not diminished with unstructured pruning**, as without a sparse tensor library, weight matrices with scattered zero *elements* must still be stored (Han et al., 2016). In contrast, structurally pruned models do not rely on such libraries at test-time since non-zero *units* can simply be spliced together.

Pruning Methodology. The structured pruning process is depicted in Figure 1. In each pruning iteration, we rank each filter by its ℓ_2 norm, greedily remove filters with the smallest magnitudes, and splice together non-zero filters in the underlying weight matrix. The deletion of a single filter results in one less output channel, implying we can also remove the corresponding input channel of the subsequent linear layer with a similar splicing operation. Repetitions of this process result in an objectively smaller model because of reductions in the convolutional and linear layer weight matrices. Furthermore, this process does not lead to irregular sparsity patterns, resulting in a general speedup on all hardware platforms.

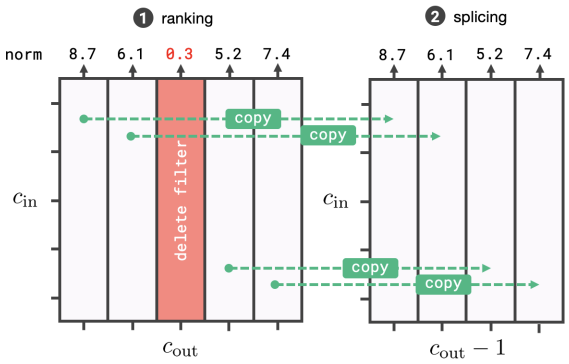

Figure 1: Structured pruning of convolutional models by (1) ranking filters by their ℓ_2 norm, then (2) splicing out the lowest norm filter, resulting in a successively smaller weight matrix. Because each filter convolves input filters c_{in} into one output filter c_{out}, removing a single filter results in $c_{\text{out}} - 1$ output channels.

The heuristic function for ranking filters and whether to re-train the model after a pruning step are important hyperparameters. We experimented with both ℓ_1 and ℓ_2 norms for selecting filters, and found that ℓ_2 slightly outperforms ℓ_1. More complicated heuristic functions, such as deriving filter importance according to gradient saliency (Persand et al., 2020), can also be dropped into our pipeline without modification.

One-Shot vs. Iterative Pruning. Furthermore, when deciding to re-train the model, we experiment with one-shot and iterative pruning (Frankle and Carbin, 2019). One-shot pruning involves repeatedly deleting filters until reaching a desired sparsity level without re-training, whereas iterative pruning interleaves pruning and re-training, such that the model is re-trained to convergence after each pruning step. These re-training steps increase overall training time, but implicitly help the model "reconstruct" deleted filter(s), resulting in significantly better performance. During test-time, however, the pruned model uses significantly fewer resources, as we demonstrate in our acceleration experiments.

5 Tasks and Datasets

We build convolutional models for intent detection and slot filling, two popular natural language understanding tasks in the task-oriented dialog stack. Intent detection is a multi-class classification problem, whereas slot filling is a sequence labeling problem. Formally, given utterance tokens $\mathbf{w} = (w_1, \cdots, w_n)$, models induce a joint distribution $P(y^*_{\text{intent}}, \mathbf{y}^*_{\text{slot}} | \mathbf{w})$ over an intent label y^*_{intent}

Models	ATIS		Snips	
	Intent	Slot	Intent	Slot
Baselines				
Slot-Gated RNN	94.10	95.20	97.00	88.80
Stack Propagation	96.90	95.90	98.00	94.20
DistilBERT (66M)	96.98	95.44	97.94	94.59
BERT (110M)	97.16	96.02	98.26	95.05
Method: No Compression				
Single-Task	94.94	94.01	96.54	85.06
Multi-Task (195K/174K)	94.98	94.30	96.97	84.38
Method: Structured Pruning				
Single-Task	95.45	94.61	96.94	85.11
Multi-Task (97K/87K)	95.39	94.42	97.17	83.81

Table 1: Intent accuracy and slot F1 of baseline models (Goo et al., 2018; Qin et al., 2019; Sanh et al., 2019; Devlin et al., 2019) and our systems on ATIS and Snips. We experiment with single-task and multi-task models. Number of model parameters are shown in parentheses where applicable; multi-task models use the format (ATIS/Snips).

and slot labels $\mathbf{y}^*_{\text{slot}} = (\mathbf{y}^{*(1)}_{\text{slot}}, \cdots, \mathbf{y}^{*(n)}_{\text{slot}})$. These models are typically multi-task: intent and slots predictions are derived with task-specific heads but share a common representation (Liu and Lane, 2016). However, since the intent and slots of an utterance are independent, we can also learn single-task models, where an intent model optimizes $P(y^*_{\text{intent}}|\mathbf{w})$ and a slot model optimizes $P(\mathbf{y}^*_{\text{slot}}|\mathbf{w})$. We experiment with both approaches, although our ultimate compressed model is multi-tasked as aligned with on-device use cases.

Following previous work, we evaluate on ATIS (Hemphill et al., 1990) and Snips (Coucke et al., 2018), both of which are single-turn dialog benchmarks with intent detection and slot filling. ATIS has 4,478/500/893 train/validation/test samples, respectively, with 21 intents and 120 slots. Snips has 13,084/700/700 samples with 7 intents and 72 slots. Our setup follows the same preparation as Zhang et al. (2019).

6 Experiments and Results

We evaluate the performance, compression, and acceleration of our structured pruning approach against several baselines. Note that we do not employ post-hoc compression methods like quantization (Guo, 2018), as they are orthogonal to our core method, and can be utilized at no additional cost to further improve performance on-device.

Params	CR (%)	Pruning		Distillation	
		Intent	Slot	Intent	Slot
195K	0%	94.98	94.30	93.84	94.12
156K	20%	95.39	94.19	94.85	94.22
117K	40%	95.03	94.14	94.51	94.13
78K	60%	95.10	94.12	92.27	94.32
39K	80%	94.40	93.91	90.48	94.05
19K	90%	92.23	93.20	78.28	92.46
9K	95%	88.35	92.19	70.89	89.54
2K	99%	79.49	87.17	70.89	64.75

Table 2: ATIS performance of multi-task models compressed with structured pruning (ours) and knowledge distillation (Hinton et al., 2015) as the compression rate (CR; %) increases. We report intent accuracy and slot F1. Darker shades of red indicate higher absolute performance drops with respect to 100%.

6.1 Benchmark Results

We experiment with both single-task and multi-task models, with and without structured pruning, on ATIS and Snips. The results are displayed in Table 1. Our multi-task model with structured pruning, even with over a 50% reduction in parameters, performs on par with our NO COMPRESSION baselines. On ATIS, our model is comparable to SLOT-GATED RNN (Goo et al., 2018) and is only about 2% worse in accuracy/F1 than BERT. However, we note that our model's slot F1 severely drops off on Snips, possibly because it is a much larger dataset spanning a myriad of domains. Whether our pre-trained embeddings have sufficient explanatory power to scale past common utterances is an open question.

Furthermore, to approximate what information is lost after compression, we analyze which samples' predictions flip from correct to incorrect after structured pruning. We observe that sparser models tend to prefer larger classes; for example, in slot filing, tags are often mislabeled as "outside" in IOB labeling (Tjong and Sang, 2000). This demonstrates a trade-off between preserving non-salient features that work on average for all classes or salient features that accurately discriminate between the most prominent classes. Our model falls on the right end of this spectrum, in that it greedily de-prioritizes representations for inputs that do not contribute as much to aggregate dataset log likelihood.

6.2 Comparison with Distillation

In addition, we compare structured pruning with knowledge distillation, a popular compression technique where a small, student model learns from a large, teacher model by minimizing the KL diver-

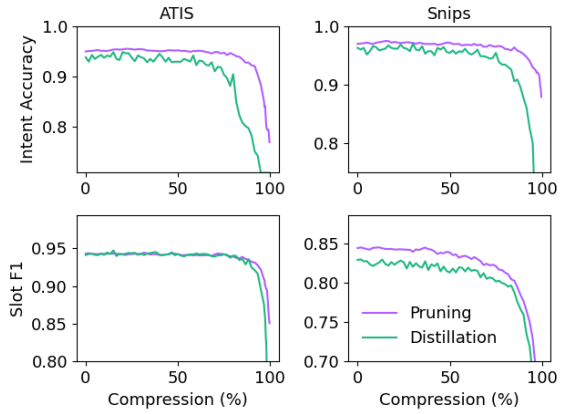

Figure 2: Performance-compression tradeoff curves on ATIS and Snips, comparing multi-task models compressed with structured pruning (ours) and knowledge distillation (Hinton et al., 2015). Pruning curves denote the mean of five compression runs with random restarts. Note that the y-axis ticks are *not* uniform across graphs.

gence between their output distributions (Hinton et al., 2015). Using a multi-task model on ATIS, we compress it with structured pruning and distillation, then examine its performance at varying levels of compression. The results are shown in Table 2. Distillation achieves similar results as structured pruning with 0-50% sparsity, but its performance largely drops off after 80%. Surprisingly, even with extreme compression (99%), structured pruning is about 10% and 20% better on intents and slots, respectively.

Our results show that, in this setting, the iterative refinement of a sparse topology admits an easier optimization problem; learning a smaller model directly is not advantageous, even when it is supervised with a larger model. Furthermore, the iterative nature of structured pruning means it is possible to select a model that optimizes a particular performance-compression trade off along a sparsity curve, as shown in Figure 2. To do the same with distillation requires re-training for a target compression level each time, which is intractable with a large set of hyperparameters.

6.3 Acceleration Experiments

Lastly, to understand how our multi-task model with structured pruning performs without significant computational resources, we benchmark its test-time performance on a CPU and GPU. Specifically, we measure several models' inference times on ATIS and Snips (normalized by the total number of test samples) using an Intel Xeon E3-1270 v3 CPU and NVIDIA GTX 1080-TI GPU. Results are shown in Table 3. Empirically, we see that our pruned model results in significant speedups without a GPU compared to both a distilled model and BERT. DistilBERT, which is a strong approximation of BERT, is still 63× slower than our model. We expect that latency disparities on weaker CPUs will be even more extreme, therefore selecting a model that maximizes both task performance and resource-efficiency will be an important consideration for practitioners.

System	ATIS		Snips	
	CPU ↓	GPU ↓	CPU ↓	GPU ↓
DistilBERT	22.15 ms	1.87 ms	21.81 ms	1.76 ms
BERT	43.19 ms	2.80 ms	43.04 ms	2.72 ms
Pruning	**0.35** ms	**0.37** ms	**0.33** ms	**0.36** ms
Distillation	0.40 ms	0.37 ms	0.38 ms	0.37 ms

Table 3: Average CPU and GPU inference time (in milliseconds) of baselines (Sanh et al., 2019; Devlin et al., 2019) and our multi-task models on ATIS and Snips.

7 Conclusion

In this work, we show that structurally pruned convolutional models achieve competitive performance on intent detection and slot filling at only a fraction of the size of state-of-the-art models. Our method outperforms popular compression methods, such as knowledge distillation, and results in large CPU speedups compared to BERT and DistilBERT.

Acknowledgments

Thanks to our anonymous reviewers for their helpful comments and feedback.

References

Afzal Ahmad and Muhammad Adeel Pasha. 2020. FFConv: An FPGA-Based Accelerator for Fast Convolution Layers in Convolutional Neural Networks. *ACM Transactions on Embedded Computing Systems*.

Sajid Anwar, Kyuyeon Hwang, and Wonyong Sung. 2017. Structured Pruning of Deep Convolutional Neural Networks. *ACM Journal on Emerging Technologies in Computing Systems (JETC)*.

Shaojie Bai, J. Zico Kolter, and Vladlen Koltun. 2018. An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling. *arXiv preprint arXiv:1803.01271*.

Daniel G. Bobrow, Ronald M. Kaplan, Martin Kay, Donald A. Norman, Henry S. Thompson, and Terry Winograd. 1977. GUS, A Frame-Driven Dialog System. *Artificial Intelligence*.

Yu Cheng, Duo Wang, Pan Zhou, and Tao Zhang. 2017. A Survey of Model Compression and Acceleration for Deep Neural Networks. *arXiv preprint arXiv:1710.09282*.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel P. Kuksa. 2011. Natural Language Processing (Almost) from Scratch. *Journal of Machine Learning Research (JMLR)*.

Alice Coucke, Alaa Saade, Adrien Ball, Théodore Bluche, Alexandre Caulier, David Leroy, Clément Doumouro, Thibault Gisselbrecht, Francesco Caltagirone, Thibaut Lavril, Maël Primet, and Joseph Dureau. 2018. Snips Voice Platform: an embedded Spoken Language Understanding system for private-by-design voice interfaces. *arXiv preprint arXiv:1805.10190*.

Shrey Desai, Geoffrey Goh, Arun Babu, and Ahmed Aly. 2020. Lightweight Convolutional Representations for On-Device Natural Language Processing. *arXiv preprint arXiv:2002.01535*.

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 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*.

Jonathan Frankle and Michael Carbin. 2019. The Lottery Ticket Hypothesis: Finding Sparse, Trainable Neural Networks. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Jianfeng Gao, Michel Galley, and Lihong Li. 2018. Neural Approaches to Conversational AI. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (NAACL): Tutorial Abstracts*.

Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N. Dauphin. 2017. Convolutional Sequence to Sequence Learning. In *Proceedings of the International Conference on Machine Learning (ICML)*.

Chih-Wen Goo, Guang Gao, Yun-Kai Hsu, Chih-Li Huo, Tsung-Chieh Chen, Keng-Wei Hsu, and Yun-Nung Chen. 2018. Slot-Gated Modeling for Joint Slot Filling and Intent Prediction. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*.

Mitchell A Gordon, Kevin Duh, and Nicholas Andrews. 2020. Compressing BERT: Studying the Effects of Weight Pruning on Transfer Learning. *arXiv preprint arXiv:2002.08307*.

Yunhui Guo. 2018. A Survey on Methods and Theories of Quantized Neural Networks. *arXiv preprint arXiv:1808.04752*.

Sonal Gupta, Rushin Shah, Mrinal Mohit, Anuj Kumar, and Mike Lewis. 2018. Semantic Parsing for Task Oriented Dialog using Hierarchical Representations. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Song Han, Huizi Mao, and William J. Dally. 2016. Deep Compression: Compressing Deep Neural Network with Pruning, Trained Quantization and Huffman Coding. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Charles T. Hemphill, John J. Godfrey, and George R. Doddington. 1990. The ATIS Spoken Language Systems Pilot Corpus. In *Speech and Natural Language: Proceedings of a Workshop Held at Hidden Valley, Pennsylvania*.

Matthew Henderson. 2015. Machine Learning for Dialog State Tracking: A Review. In *Proceedings of The First International Workshop on Machine Learning in Spoken Language Processing*.

Geoffrey E. Hinton, Oriol Vinyals, and Jeffrey Dean. 2015. Distilling the Knowledge in a Neural Network. *arXiv preprint arXiv:1503.02531*.

Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. 2020. TinyBERT: Distilling BERT for Natural Language Understanding. *arXiv preprint arXiv:1909.10351*.

Yoon Kim. 2014. Convolutional Neural Networks for Sentence Classification. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. *Proceedings of the International Conference on Learning Representations (ICLR)*.

Zornitsa Kozareva and Sujith Ravi. 2018. Fast & Small On-device Neural Networks for Short Text Natural Language Processing. In *Proceedings of the NeurIPS Workshop on Machine Learning on the Phone and other Consumer Devices (MLPCD)*.

Yann LeCun, John S. Denker, and Sara A. Solla. 1990. Optimal Brain Damage. In *Proceedings of the Conference on Neural Information Processing Systems (NeurIPS)*.

Hao Li, Asim Kadav, Igor Durdanovic, Hanan Samet, and Hans Peter Graf. 2017. Pruning Filters for Efficient ConvNets. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Ziheng Lin, Yan Gu, and Samarjit Chakraborty. 2010. Tuning Machine-Learning Algorithms for Battery-Operated Portable Devices. In *Proceedings of the Asia Information Retrieval Symposium (AIRS)*.

Bing Liu and Ian Lane. 2016. Attention-Based Recurrent Neural Network Models for Joint Intent Detection and Slot Filling. In *Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH)*.

Andrea K. McIntosh, Safwat Hassan, and Abram Hindle. 2018. What can Android mobile app developers do about the energy consumption of machine learning? *Empirical Software Engineering*.

Pavlo Molchanov, Stephen Tyree, Tero Karras, Timo Aila, and Jan Kautz. 2016. Pruning Convolutional Neural Networks for Resource Efficient Inference. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global Vectors for Word Representation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Kaveena Persand, Andrew Anderson, and David Gregg. 2020. Composition of Saliency Metrics for Channel Pruning with a Myopic Oracle. *arXiv preprint arXiv:2004.03376*.

Libo Qin, Wanxiang Che, Yangming Li, Haoyang Wen, and Ting Liu. 2019. A Stack-Propagation Framework with Token-Level Intent Detection for Spoken Language Understanding. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing and the International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*.

Abhinav Rastogi, Xiaoxue Zang, Srinivas Sunkara, Raghav Gupta, and Pranav Khaitan. 2020. Towards Scalable Multi-domain Conversational Agents: The Schema-Guided Dialogue Dataset. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*.

Sujith Ravi and Zornitsa Kozareva. 2018. Self-Governing Neural Networks for On-Device Short Text Classification. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

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*.

Nitish Srivastava, Geoffrey E. Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research (JMLR)*.

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*.

Erik F. Tjong and Kim Sang. 2000. Transforming a chunker to a parser. In *Proceedings of the Meeting of Computational Linguistics in the Netherlands (CLIN)*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *Proceedings of the Conference on Neural Information Processing Systems (NeurIPS)*.

Yu Wang, Yilin Shen, and Hongxia Jin. 2018. A Bi-Model Based RNN Semantic Frame Parsing Model for Intent Detection and Slot Filling. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*.

Krzysztof Wróbel, Marcin Pietroń, Maciej Wielgosz, Michal Karwatowski, and Kazimierz Wiatr. 2018. Convolutional neural network compression for natural language processing. *arXiv preprint arXiv:1805.10796*.

Zhao Yan, Nan Duan, Peng Chen, Ming Zhou, Jianshe Zhou, and Zhoujun Li. 2017. Building Task-Oriented Dialogue Systems for Online Shopping. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*.

Hong-Jun Yoon, Sarah Robinson, J. Blair Christian, John X. Qiu, and Georgia D. Tourassi. 2018. Filter pruning of Convolutional Neural Networks for text classification: A case study of cancer pathology report comprehension. In *Proceedings of the IEEE EMBS International Conference on Biomedical & Health Informatics (BHI)*.

Zhou Yu, Ziyu Xu, Alan W Black, and Alexander Rudnicky. 2016. Strategy and Policy Learning for Non-Task-Oriented Conversational Systems. In *Proceedings of the Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*.

Ofir Zafrir, Guy Boudoukh, Peter Izsak, and Moshe Wasserblat. 2019. Q8BERT: Quantized 8Bit BERT. *arXiv preprint arXiv:1910.06188*.

Chenwei Zhang, Yaliang Li, Nan Du, Wei Fan, and Philip Yu. 2019. Joint Slot Filling and Intent Detection via Capsule Neural Networks. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*.

Zheng Zhang, Ryuichi Takanobu, Minlie Huang, and Xiaoyan Zhu. 2020. Recent Advances and Challenges in Task-oriented Dialog System. *arXiv preprint arXiv:2003.07490*.