

Knowledge Distillation Transfer Sets and their Impact on Downstream NLU Tasks

Charith Peris *

Amazon, Cambridge, USA
perisc@amazon.com

Lizhen Tan

Amazon, Cambridge, USA
ltn@amazon.com

Thomas Gueudre

Amazon, Turin, Italy
tgueudre@amazon.it

Turan Gojayevev

Amazon, Berlin, Germany
tgojayevev@amazon.de

Pan Wei

Amazon, Cambridge, USA
panwei@amazon.com

Gokmen Oz

Amazon, Cambridge, USA
ogokmen@amazon.com

Abstract

Teacher-student knowledge distillation is a popular technique for compressing today’s prevailing large language models into manageable sizes that fit low-latency downstream applications. Both the teacher and the choice of transfer set used for distillation are crucial ingredients in creating a high quality student. Yet, the generic corpora used to pre-train the teacher and the corpora associated with the downstream target domain are often significantly different, which raises a natural question: *should the student be distilled over the generic corpora, so as to learn from high-quality teacher predictions, or over the downstream task corpora to align with finetuning?* Our study investigates this trade-off using Domain Classification (DC) and Intent Classification/Named Entity Recognition (ICNER) as downstream tasks. We distill several multilingual students from a larger multilingual LM with varying proportions of generic and task-specific datasets, and report their performance after finetuning on DC and ICNER. We observe significant improvements across tasks and test sets when only task-specific corpora is used. We also report on how the impact of adding task-specific data to the transfer set correlates with the similarity between generic and task-specific data. Our results clearly indicate that, while distillation from a generic LM benefits downstream tasks, students learn better using target domain data even if it comes at the price of noisier teacher predictions. In other words, target domain data still trumps teacher knowledge.

1 Introduction

In the recent past, large language models (LMs; BERT-Large, Devlin et al., 2019; GPT-2, Radford et al., 2019; T5, Raffel et al., 2020) pretrained in a self-supervised manner on massive web corpora have consistently shown state-of-the-art per-

formance for multiple natural language understanding (NLU) tasks. Therefore, it is no surprise that these models are of much interest for virtual assistants such as Amazon Alexa, Apple Siri, and Google Assistant. Some studies have shown that these large models trained on generic corpora seem to be more robust to data distributional shifts, relying less on domain-specific training data to perform well (Brown et al., 2020).

Since large models cannot be directly used for low-latency applications on devices with limited computing capacity, many techniques have been developed to compress them in size. Knowledge distillation (referred to simply as distillation hereafter; Hinton et al., 2015), has shown promising results, especially at the high compression rates typically required in NLU (Jiao et al., 2020, Soltan et al., 2021). In this paradigm, lightweight models referred to as students, are trained to mimic the teacher predictions over a transfer set (Hinton et al., 2015). When the pretraining and task-specific corpora have significantly different distributions, as is often the case, the choice of data for the transfer set can be ambiguous. On the one hand, using pretraining corpora in the transfer set ensures high quality teacher predictions that are important for effective distillation. On the other, using the downstream corpora, although it might cause noisier teacher predictions, ensures the adaptation of the student to its final use case.

To investigate this trade-off, we present a set of experiments where we distill several multilingual students from a large multilingual teacher LM trained using a masked language modeling (MLM) objective. We perform the distillations using transfer sets that comprise of generic and task-specific data in varying proportions. The students are then finetuned and evaluated on two downstream NLU tasks of interest: a Domain Classification (DC) task and a joint Intent Classification/Named Entity Recognition (ICNER) task. For each input utter-

* Corresponding Author

ance DC predicts the relevant domain (Books, Music, Shopping, etc.), IC identifies the user’s intent (find a book, play a song, buy an item, etc.) and NER extracts the entities in the utterance (dates, names, locations, etc.).

Our contributions: (1) We confirm for our setup that model preparation via distillation from a larger LM is more beneficial for downstream task performance when compared to encoder training from scratch. (2) We show that the largest improvements are seen when using only the downstream task’s unlabelled data during the distillation process. Even though teacher predictions are expected to be noisy over data that is different from pre-training corpora, our results clearly indicate that students learn best in this setting. (3) Because our ICNER corpora is divided per domain, we are also able to provide a finer-grained analysis of the impact of corpora similarity on downstream results. (4) Finally, we also confirm that further adaptation of the teacher to the target-domain data, results in improved student performance across tasks.

2 Relevant Work

Building models with inference speeds that are suitable for production systems is of utmost importance in the industrial setting. Therefore techniques for model compression (quantization [Gong et al., 2014](#); pruning redundant connections [Han et al., 2015](#)) have been active research topics, with distillation ([Romero et al., 2015](#), [Hinton et al., 2015](#), [Jiao et al., 2020](#)) showing much promise for NLU models ([Sanh et al., 2019](#)). Distillation processes and their data have evolved over the past few years. In the teacher-student framework proposed by [Hinton et al. \(2015\)](#), they recommend using the original pretraining set as the transfer set. [Jiao et al. \(2020\)](#) proposes a more complex two-stage process with generic and task-specific distillation phases, each with their own data sets, designed to augment the performance of the final model towards the task at hand.

Our work is focused on exploring how varying proportions of generic and task-specific data within the transfer set of a single distillation process impacts downstream NLU performance. Since our scope does not include optimizing the distillation process itself, we use a cheaper alternative to [Jiao et al. \(2020\)](#), via a single-stage distillation setup to conduct our exploration (see Section [A.3](#) for details).

[Gururangan et al. \(2020\)](#) showed for the *pretraining* phase, that continued domain-adaptive and task-adaptive pretraining using the downstream task’s unlabeled data can improve performance. Our work presents similar results for the distillation phase.

3 Data

3.1 Distillation data

For distillation, we created the transfer sets by mixing two types of data with different distributions:

- **Generic data:** This data set consisted of Wikipedia and Common Crawl processed by an in-house tokenizer.
- **Task-specific data:** This in-house data set comprised of de-identified utterances from a voice assistant across domains of interest. The text data collected here was the output of an Automatic Speech Recognition (ASR) model, which assigned a confidence score per utterance. In order to retain only the highest quality data, we filtered it by an ASR score threshold. The data was de-identified, prior to use.

Our distilled students were trained as part of a larger program resulting in a collection of nine European and Indic languages being used for distillation. The language list and counts are shown in [Table A1](#).

We built transfer sets that had three ratios of generic to task-specific data: (1) generic-only (baseline) (2) 7:3 generic to task-specific, to mimic the commonly encountered low task-specific data setting and (3) task-specific-only. To have a comparable distribution of data from each language, we created samples of equal size for each language using either generic only, task-specific only or combining both the generic and the task-specific data based on the targeted ratio. Upsampling is used when a source data set contains a number less than the required number. The 7:3 ratio consisted of Wikipedia, Common Crawl and task-specific data upsampled to counts of 35M, 35M and 30M respectively, for each language. For two languages Indian-English and Marathi, where some data constituents were unobtainable, available data was used in proportion (see [Table A1](#)). Once the data sets were created with the targeted mixing ratio, they were split into train and validation sets with a ratio of 0.995:0.005 and then used in the transfer sets.

3.2 Data for downstream tasks

We evaluated our multilingual distilled students in the context of two commonly utilized NLU tasks of interest, DC and ICNER. We limit the scope of our evaluation to just four languages German, French, Italian and Spanish. Our finetuning data consisted of 26 domains (see fractional utterance counts in Table A4) across each language, with each domain comprising a set of intents (similar to Su et al. 2018). As with the task-specific data used in our transfer sets, this data has also been de-identified prior to use.

It is important to note that, although collected over non-overlapping time intervals (and thus consisting of different absolute counts), the finetuning data was from the same distribution as the task-specific data described in Section 3.1. We sampled the finetuning data so as to have equal counts across each domain in all four languages (see Appendix A.1 for the evaluation data set sampling strategy). We then combined all languages and split the data into proportions of 80:10:10 for train, validation and test, respectively.

For the DC task, we classified the input utterances into one of the 26 domains. Therefore, the DC model is trained using the combined training data from the four languages across all domains and is tested on language-specific test data sets. For the joint ICNER task, we classified each utterance within a domain to its corresponding intent and also recognized its named-entities. For this task, we trained a model per domain, using the combined training data from the four languages for that domain. The model was evaluated using language-specific test data sets for that domain. We present results on two types of test sets. *test* comprises of the full test set obtained from the split above while *tail_test* is the subset of data points within *test* that have a frequency of occurrence less than or equal to 3. The relative data proportions used can be found in the Appendix (Table A4).

4 Models

Figure 1 shows a schematic of the models and experimental setup described in this section.

4.1 Distilled students and baselines

We use a 170 million parameter teacher (170M-teacher) that was prepared using Wikipedia, Common Crawl and mC4 (Xue et al., 2021) data. See Appendix A.2 for details on teacher prepa-

ration. From this teacher, we distilled a total of five students. We use our three transfer sets described in Section 3.1, i.e. (1) generic-only (2) 7:3 (generic:task-specific) and (3) task-specific-only, to distill the first three students. We refer to the student distilled using (1) as the generic-distilled baseline. The latter two are referred to as experiment-7:3 and experiment-task-specific-only; the naming aligned with the transfer set used. In addition to these, we create another two students where the teacher was finetuned using an MLM task before being used for distillation. In each case, the teacher was finetuned for 15625 steps using the same transfer set that was used for the subsequent distillation. We refer to these two students as experiment-7:3-FT and experiment-task-specific-only-FT. The teacher finetuning was run on a p3.16X instance with an average run time of approximately 45 hours. We collectively refer to all distilled students that are not a baseline as experimental students.

The architectures of our teacher and students are as follows. As in the paper by Devlin et al. (2019), we denote the number of layers (i.e., Transformer blocks) as L, the hidden size as H, and the number of self-attention heads as A.

- 170M-teacher: L=16, H=1024, A=16, feed-forward/filter size=3072, total parameters=170M
- Students: L=4, H=768, A=16, feed-forward/filter size=1200, total parameters=17M

For a description of the distillation setup, see Appendix A.3. Distillation was run for 1 epoch with each student extracted at 78125 steps, which equates to approximately 80M data points seen. We ran distillation on a single p3.16X instance utilizing 8 GPUs with batch-size of 2 and gradient accumulation at every 64 steps. The average run time was approximately 195 hours. Note that each distillation run used only a sample of the full data set mentioned in Section 3.1, determined by the step count. However, since the data is sampled uniformly, the language ratios and the generic:task-specific data ratio stays consistent during training.

In addition to the distilled baseline, we also created another baseline (*without distillation*) that was directly pretrained using the generic-only data. The architecture and size of this baseline was identical to that of the distilled students and it is referred to, here onward, as the directly-pretrained baseline.

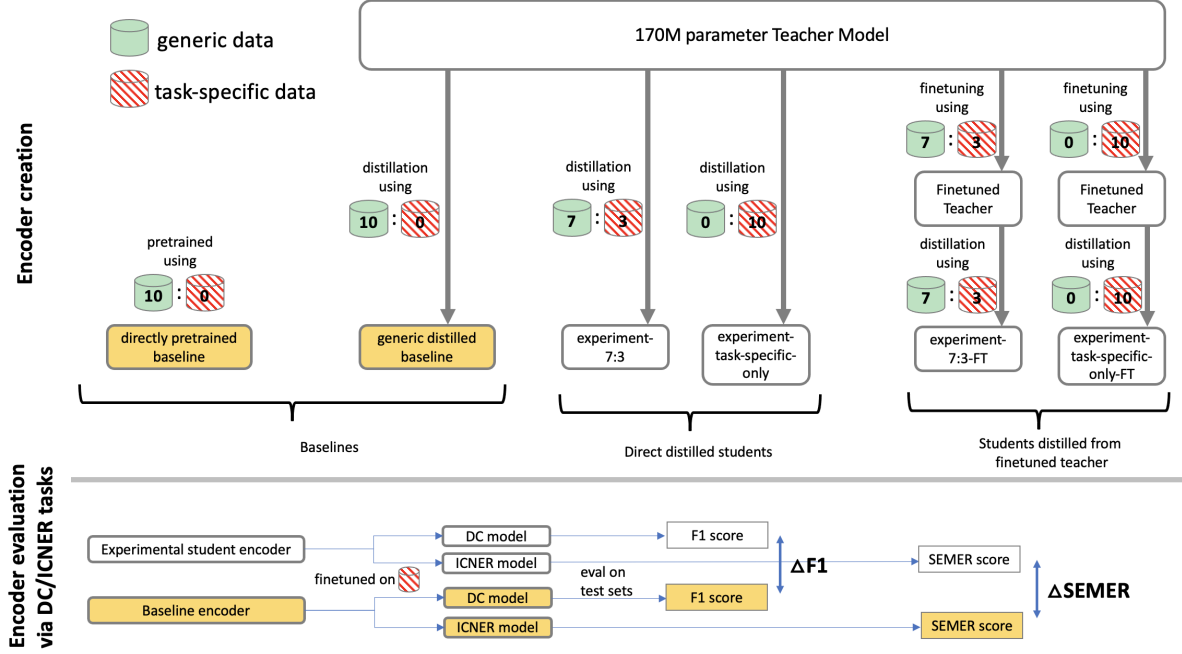


Figure 1: A schematic of the models that we present in this paper and how they are evaluated.

We used this baseline to observe performance differences between models that use students distilled from the large teacher and those that use a directly pretrained encoder.

4.2 DC and ICNER models

In order to evaluate the impact of the different transfer sets on our targeted downstream NLU tasks, we finetune the experimental students and baselines toward DC and ICNER tasks. Each DC model consisted of an encoder, embedding and positional embedding obtained from an experimental student or baseline combined with a decoder consisting of an MLP classifier for domain prediction with layer size 128, dropout set at 0.1 and ReLU activation. Each ICNER model consisted of the same encoder, embedding and positional embeddings used for the corresponding DC model with an MLP classifier output layer for the IC task with layer size 128, dropout set at 0.1 and ReLU activation and a CRF sequence-labeler output layer for the NER task with layer size 256, dropout set at 0.1 and GeLU activation. We trained each DC model for 1 epoch and each ICNER model for 4 epochs.

Evaluation: The DC performance was evaluated using the F1 score while the ICNER performance was evaluated using the Semantic Error Rate (SemER; Su et al., 2018, Varada et al., 2020, Peris et al., 2020). The definition of SemER is

$$SemER = \frac{(D + I + S)}{(C + D + S)} \quad (1)$$

where D (deletion), I (insertion), S (substitution), C (correct slots). The Intent was treated as a slot in this metric, and the Intent error was considered as a substitution.

5 Experiments

5.1 Experimental results

In this section, note that *model* refers a model that uses an experimental student or baseline encoder and has been finetuned towards a DC or ICNER task. Experimental models comprise of experimental student encoders and baseline models comprise of baseline encoders (see lower panel in Figure 1).

We used data across 26 domains to train and evaluate the DC and ICNER models (see Section 3.2). We compare the performance of each experimental model against the two baseline models (see Section 4.1). The improvements we quote in this section are $\Delta F1$ (\uparrow) (*higher is better*) and $\Delta SemER$ (\downarrow) (*lower is better*; we use the weighted average of SemER across all domains) for DC and ICNER respectively, measured against the baseline models (Tables 1, 2, A2, A3).

The results in Tables 1 and 2 show that in general for both DC and ICNER tasks, all experimental students distilled with a mix of task-specific data

Distilled encoder	Baseline	Test Set	German (%)	French (%)	Italian (%)	Spanish (%)
experiment-7:3	generic distilled	test	0.19 ± 0.02	0.19 ± 0.04	0.21 ± 0.03	0.24 ± 0.03
experiment-task-specific-only	generic distilled	test	0.51 ± 0.01	0.54 ± 0.03	0.47 ± 0.03	0.55 ± 0.03
experiment-7:3-FT	generic distilled	test	0.31 ± 0.03	0.3 ± 0.02	0.31 ± 0.03	0.35 ± 0.03
experiment-task-specific-only-FT	generic distilled	test	0.69 ± 0.02	0.79 ± 0.03	0.7 ± 0.03	0.79 ± 0.02
experiment-7:3	generic distilled	tail test	0.34 ± 0.05	0.31 ± 0.09	0.42 ± 0.05	0.42 ± 0.05
experiment-task-specific-only	generic distilled	tail test	1.0 ± 0.03	1.05 ± 0.04	1.02 ± 0.06	1.07 ± 0.06
experiment-7:3-FT	generic distilled	tail test	0.56 ± 0.06	0.61 ± 0.04	0.65 ± 0.06	0.6 ± 0.07
experiment-task-specific-only-FT	generic distilled	tail test	1.38 ± 0.05	1.51 ± 0.06	1.48 ± 0.06	1.51 ± 0.05

Table 1: Relative DC $\Delta F1$ (\uparrow), measured against the generic distilled baseline for each experimental student (positive is better). We run three iterations of each experimental student and show the percentage change of their means and its standard deviation.

Distilled encoder	Baseline	Test Set	German (%)	French (%)	Italian (%)	Spanish (%)
experiment-7:3	generic distilled	test	-0.55 ± 0.09	-0.31 ± 0.07	-0.17 ± 0.12	-0.17 ± 0.09
experiment-task-specific-only	generic distilled	test	-1.25 ± 0.07	-0.81 ± 0.07	-0.56 ± 0.08	-1.3 ± 0.09
experiment-7:3-FT	generic distilled	test	-0.83 ± 0.09	-0.49 ± 0.13	-0.06 ± 0.14	-0.58 ± 0.09
experiment-task-specific-only-FT	generic distilled	test	-1.57 ± 0.15	-1.18 ± 0.07	-0.6 ± 0.25	-1.26 ± 0.04
experiment-7:3	generic distilled	tail test	-0.49 ± 0.07	-0.31 ± 0.06	-0.16 ± 0.09	-0.23 ± 0.11
experiment-task-specific-only	generic distilled	tail test	-1.19 ± 0.05	-0.86 ± 0.07	-0.67 ± 0.08	-1.44 ± 0.09
experiment-7:3-FT	generic distilled	tail test	-0.83 ± 0.09	-0.52 ± 0.1	-0.13 ± 0.09	-0.65 ± 0.11
experiment-task-specific-only-FT	generic distilled	tail test	-1.53 ± 0.12	-1.26 ± 0.07	-0.79 ± 0.14	-1.32 ± 0.06

Table 2: Relative ICNER ΔSemER (\downarrow), measured against the generic distilled baseline for each experimental student (negative is better). As with DC, we run three iterations of the experimental students and show the percentage change of their means and its standard deviation. In calculating these percentage changes, we use the weighted average of the SemER for each domain in a given language, as the overall SemER in that language.

(30% or 100%) perform significantly better than the generic distilled baseline. We further observe that models with encoders distilled with task-specific-only data yields the best overall performance which means that, in our setup, students learn better using target-domain data even if it comes at the price of noisier teacher predictions.

For all four languages across DC and for three out of four languages across ICNER, the best performances are observed with student models that were distilled from the *finetuned* teacher. This confirms that the additional step of finetuning the teacher and adapting it to the task-specific dataset, results in students that perform better on the intended downstream tasks.

We also note that across all task, language and test set combinations, the improvements seen against the directly pretrained baseline (see Tables A2 and A3) are larger than the improvements seen against the generic distilled baseline. For our setup, this shows that distilling from a large LM can benefit downstream tasks as opposed to using a similar-sized encoder pretrained from scratch; in other words our findings suggest that it is *better to distill* than to directly pretrain. However, we note

that additional resources (in our case approximately 45 p3.16X hours) are required for this.

The *tail_test*, comprising of low frequency utterances within test, provides insights on the ability of the model to generalize to rarely seen utterances. For DC, we note that the improvements on *tail_test* are significantly larger ($\sim 2X$) than the improvements seen on the test set. This indicates that prediction on examples that appear infrequently in the task-specific data benefits more, from task-specific data being included in the distillation process.

5.2 Dataset similarity and its correlation to SemER improvements for ICNER

To further explore our conclusion that *students learn better using target-domain data* we explore how ΔSemER for each domain, correlate to the similarity of the domain’s data to the generic data. Note that, here, negative ΔSemER represents improvements of the experimental students against the generic distilled baseline while the opposite is true for positive ΔSemER . SemER results are from the *test* set.

The hypothesis here is that the more distant a domain is from the generic data, the more value we should see in adding this domain’s data to the distil-

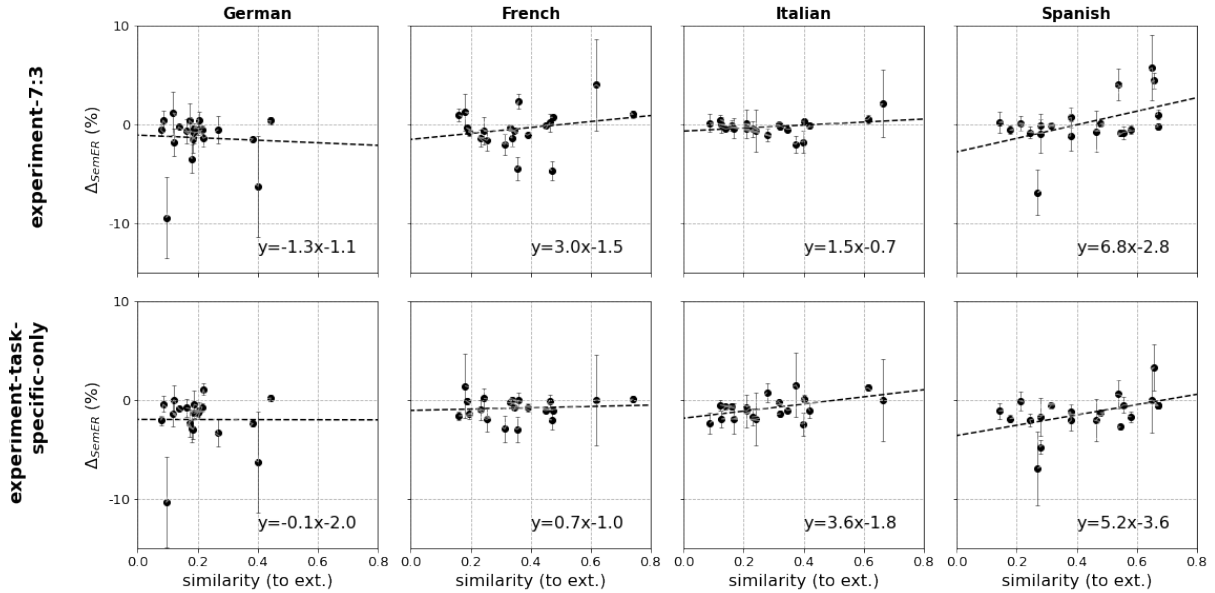


Figure 2: Cosine similarity of *tf-idf* vectors vs. change in SemER for each domain for languages German, French, Italian and Spanish. We represent only domains with >1000 test utterances to avoid noise added by smaller domains which have higher variability.

lation transfer set, even though teacher predictions might be noisy. We note here that we calculate cosine similarity on a very rudimentary corpus-level embedding (i.e. *tf-idf*) for measuring similarity, as explained below. We leave more sophisticated similarity measurements for later work.

To calculate similarity between domain-level and generic data, we use the following process. For each domain in each of the four languages, we sample up to 100K utterances. All available data is considered for domains with <100K utterances. We then sample 50K utterances each, from the Wikipedia and Common Crawl data sets of the corresponding language. We create a *tf-idf* vector for each sampled dataset and calculate their cosine similarity as a measure of dataset similarity. In order to account for any variability associated with the sampling, we repeat the process 3 times and obtain the mean similarity and the standard deviation per domain. We plot dataset similarity against ΔSemER (a single point represents one domain and a panel represents a language as seen in Figure 2). We neglect domains with lower data and thus high variability and fit a line to show how ΔSemER correlates to dataset similarity.

In Figure 2, we observe that a majority of cases (all except German) show a positive correlation. A positive correlation shows that domains that are *less* similar to Wikipedia/CommonCrawl have relatively larger improvement in SemER, when

compared to domains that are more similar to Wikipedia/CommonCrawl. This suggests that the addition of task-specific data in the distillation transfer sets helps domains that are less similar to the generic data available for distillation, even though teacher predictions on them will be more noisy.

It should be noted that the domains of the one exception, German, display low similarity values across the board unlike the other languages which show a wider spread (German has 65% of domains < 0.2 whereas French, Italian and Spanish has 23%, 31% and 12% < 0.2 respectively). The lack of domains with high similarity might explain the failure for a stable correlation to be observed in German.

6 Conclusions

We have explored how the use of transfer sets that comprise different ratios of generic to task-specific data, impacts downstream results. Encoders distilled from a large teacher perform better than ones trained from scratch, showing that it is *better to distill* than to directly pretrain, when possible. The largest benefits are shown when using the downstream task’s unlabelled data to distill, a student despite noisier teacher predictions. We also find that domains with data that are dissimilar to the generic data show greater performance improvements against a generic baseline when using a stu-

dent distilled using task-specific data. These improvements further confirm that distilling using target-domain data can be helpful for downstream performance. Finally, we show that if costs permit, teacher-adaptation to the target-domain data via finetuning can result in improved student performance across downstream tasks.

Acknowledgements

We thank Karolina Owczarzak, Fabian Triefenbach and Rahul Gupta for their support of this work and Wael Hamza for helpful discussions on the topics covered.

References

- Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. [Language models are few-shot learners](#). In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.
- 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.
- Jack G. M. FitzGerald, Shankar Ananthakrishnan, Konstantine Arkoudas, David Bernardi, Abhishek Bhagia, Claudio Delli Bovi, Jin Cao, Rakesh Chada, Amit Chauhan, Luoxin Chen, Anurag Dwarakanath, Satyam Dwivedi, Turan Gojayev, Karthik Gopalakrishnan, Thomas Gueudré, Dilek Z. Hakkani-Tür, Wael Hamza, Jonathan Hueser, Kevin Martin Jose, Haidar Khan, Bei Liu, Jianhua Lu, A. Manzotti, Pradeep Natarajan, Karolina Owczarzak, Gokmen Oz, Enrico Palumbo, Charith S. Peris, Chandan Prakash, Stephen Rawls, Andrew Rosenbaum, Anjali Shenoy, Saleh Soltan, Mukund Sridhar, Lizhen Tan, Fabian Triefenbach, Pang Wei, Haiyang Yu, Shuai Zheng, Gokhan Tur, and Premkumar Natarajan. 2022. [Alexa teacher model: Pretraining and distilling multi-billion-parameter encoders for natural language understanding systems](#). *ArXiv*, abs/2206.07808.
- Yunchao Gong, Liu Liu, Ming Yang, and Lubomir Bourdev. 2014. [Compressing deep convolutional networks using vector quantization](#).
- Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. 2020. [Don’t stop pretraining: Adapt language models to domains and tasks](#). In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8342–8360, Online. Association for Computational Linguistics.
- Song Han, Jeff Pool, John Tran, and William J. Dally. 2015. [Learning both weights and connections for efficient neural networks](#).
- Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. [Distilling the knowledge in a neural network](#).
- 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](#). In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4163–4174, Online. Association for Computational Linguistics.
- Seyed-Iman Mirzadeh, Mehrdad Farajtabar, Ang Li, Nir Levine, Akihiro Matsukawa, and Hassan Ghasemzadeh. 2019. [Improved knowledge distillation via teacher assistant](#).
- Charith Peris, Gokmen Oz, Khadige Abboud, Venkata sai Varada Varada, Prashan Wanigasekara, and Haidar Khan. 2020. [Using multiple ASR hypotheses to boost i18n NLU performance](#). In *Proceedings of the 17th International Conference on Natural Language Processing (ICON)*, pages 30–39, Indian Institute of Technology Patna, Patna, India. NLP Association of India (NLP AI).
- Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. [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. 2020. [Exploring the limits of transfer learning with a unified text-to-text transformer](#). *Journal of Machine Learning Research*, 21(140):1–67.
- Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. 2015. [Fitnets: Hints for thin deep nets](#).
- Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. [Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter](#).
- Saleh Soltan, Haidar Khan, and Wael Hamza. 2021. [Limitations of knowledge distillation for zero-shot transfer learning](#). In *Proceedings of the Second Workshop on Simple and Efficient Natural Language Processing*, pages 22–31, Virtual. Association for Computational Linguistics.

Chengwei Su, Rahul Gupta, Shankar Ananthkrishnan, and Spyridon Matsoukas. 2018. A re-ranker scheme for integrating large scale nlu models. *2018 IEEE Spoken Language Technology Workshop (SLT)*, pages 670–676.

Venkat Varada, Charith Peris, Yangsook Park, and Christopher Dipersio. 2020. [Using alternate representations of text for natural language understanding](#). In *Proceedings of the 2nd Workshop on Natural Language Processing for Conversational AI*, pages 1–10, Online. Association for Computational Linguistics.

Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. 2021. [mT5: A massively multilingual pre-trained text-to-text transformer](#). In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 483–498, Online. Association for Computational Linguistics.

A Appendix

A.1 Data for finetuning DC and ICNER models

For finetuning our distilled students for DC and ICNER, we use labelled datasets from four languages (German, French, Italian and Spanish) each consisting the same 26 domains (Table A4) and each domain supporting a set of intents (similar to Su et al. 2018). In order to have equivalent utterance counts across domains for each language, we used a stratified sampling strategy as follows. First, we ranked each language per domain based on its utterance counts. In order to prevent heavy upsampling or downsampling in any single language when creating equivalently sampled domains, we picked the language that had the second highest utterance counts in most domains (in our case French). We sampled utterances from the domains of other languages to match the domain-level utterance frequency distribution of French (i.e. random sample utterances with replacement, from each domain in each language until that number matches the utterance count of the respective domain in French). We then combined all languages and split the data into proportions of 80:10:10 for train, validation and test, respectively.

A.2 Teacher model

The 170M-teacher used in this work was, itself, a student that was distilled from a larger model with 2 billion parameters (see Stage 1 pretraining section in FitzGerald et al. (2022) for details on creation

and architecture). The 170M-teacher was distilled using a transfer set that comprised Wikipedia, Common Crawl and mC4 (Xue et al., 2021) data. Picking this intermediate-sized model helped us avoid potential performance degradation due to having too large a size gap between teacher and student (Mirzadeh et al., 2019).

A.3 Student setup

For our single-stage distillation setup, we skip the generic distillation phase done by Jiao et al. (2020) and use a non-finetuned teacher model to directly distill our students. In addition, as a sanity check, we also explore distillation from a finetuned teacher model to verify improved student performance across tasks. Similar to the hidden states based distillation followed in TinyBERT (Jiao et al., 2020), we mapped the student layers [0, 1, 2, 3] to learn from the teachers hidden layers [3, 7, 11, 15], respectively. We ignored attention based distillation (Jiao et al., 2020) since we did not observe significant improvements by using it. We also penalized the soft cross-entropy loss between the student network’s logits against the teacher’s logits, to fit the students predictions to those of the teacher as in Hinton et al. (2015). We use a MLM objective for the distillation process. In our loss, we weight the hidden layer matching, the logit matching and the MLM at a 1:2:1 ratio.

	Common Crawl (cc100)	Wikipedia	Task Specific Data
German	12,045,483	2,731,840	32,081,929
French	13,323,804	2,174,531	21,278,820
Italian	7,131,950	1,278,255	31,013,233
Spanish	11,690,123	1,825,389	22,054,722
English	14,330,660	6,360,372	19,576,081
English (IN)	-	-	27,406,082
Hindi	2,538,698	94,891	21,315,004
Tamil	919,763	66,190	-
Tamil (MT)	-	-	18,414,285
Telugu	378,812	77,179	-
Telugu (MT)	-	-	18,895,352
Marathi	263,189	21,705	-

Table A1: Raw data counts used for transfer set creation. Counts represent the number of sentences for generic data and the number of de-identified utterances for task-specific data. For task-specific data for Telugu and Tamil, machine-translated (MT) data from Indian English was used. Only task-specific data was used for Indian English because Wikipedia and Common Crawl data were not available. Only generic data was used for Marathi as the translation system used for this work did not support the language as yet.

Distilled encoder	Baseline	Test Set	German (%)	French (%)	Italian (%)	Spanish (%)
experiment-7:3	directly pretrained	test	0.31 ± 0.02	0.19 ± 0.04	0.32 ± 0.01	0.28 ± 0.03
experiment-task-specific-only	directly pretrained	test	0.63 ± 0.01	0.54 ± 0.02	0.57 ± 0.01	0.6 ± 0.02
experiment-7:3-FT	directly pretrained	test	0.42 ± 0.03	0.3 ± 0.02	0.41 ± 0.01	0.4 ± 0.02
experiment-task-specific-only-FT	directly pretrained	test	0.81 ± 0.01	0.79 ± 0.03	0.8 ± 0.02	0.83 ± 0.01
experiment-7:3	directly pretrained	tail test	0.57 ± 0.04	0.36 ± 0.08	0.57 ± 0.01	0.45 ± 0.03
experiment-task-specific-only	directly pretrained	tail test	1.23 ± 0.02	1.1 ± 0.02	1.17 ± 0.03	1.09 ± 0.04
experiment-7:3-FT	directly pretrained	tail test	0.78 ± 0.05	0.66 ± 0.02	0.8 ± 0.03	0.62 ± 0.06
experiment-task-specific-only-FT	directly pretrained	tail test	1.61 ± 0.04	1.56 ± 0.05	1.63 ± 0.03	1.54 ± 0.04

Table A2: Relative DC $\Delta F1$ (\uparrow) measured against the directly pretrained baseline for each experimental student (positive is better)

Distilled encoder	Baseline	Test Set	German (%)	French (%)	Italian (%)	Spanish (%)
experiment-7:3	directly pretrained	test	-1.73 ± 0.06	-0.7 ± 0.04	-2.3 ± 0.11	-2.44 ± 0.09
experiment-task-specific-only	directly pretrained	test	-2.42 ± 0.02	-1.19 ± 0.04	-2.69 ± 0.07	-3.54 ± 0.08
experiment-7:3-FT	directly pretrained	test	-2.01 ± 0.06	-0.88 ± 0.12	-2.2 ± 0.13	-2.84 ± 0.09
experiment-task-specific-only-FT	directly pretrained	test	-2.74 ± 0.13	-1.57 ± 0.05	-2.73 ± 0.24	-3.5 ± 0.04
experiment-7:3	directly pretrained	tail test	-1.72 ± 0.05	-0.7 ± 0.05	-2.62 ± 0.09	-2.62 ± 0.09
experiment-task-specific-only	directly pretrained	tail test	-2.41 ± 0.02	-1.24 ± 0.05	-3.11 ± 0.07	-3.8 ± 0.07
experiment-7:3-FT	directly pretrained	tail test	-2.05 ± 0.08	-0.9 ± 0.09	-2.59 ± 0.08	-3.03 ± 0.09
experiment-task-specific-only-FT	directly pretrained	tail test	-2.74 ± 0.11	-1.64 ± 0.05	-3.23 ± 0.13	-3.68 ± 0.03

Table A3: Relative ICNER ΔSemER (\downarrow) measured against the directly pretrained baseline for each experimental student (negative is better). In calculating these percentage changes, we use the weighted average of the SemER for each domain in a given language, as the overall SemER in that language.

Table A4: Fractions of finetuning data used per domain. Note that the fraction in each cell represents the utterance count for that partition, for that domain, as a fraction of the total utterance count in that language. As mentioned in Section 3.2, these fractions are not based on the counts in Table A1

	train				validation				test				tail_test			
	German	French	Italian	Spanish	German	French	Italian	Spanish	German	French	Italian	Spanish	German	French	Italian	Spanish
	Domain 1	0.014%	0.014%	0.014%	0.014%	0.002%	0.002%	0.002%	0.002%	0.002%	0.002%	0.002%	0.002%	0.002%	0.002%	0.002%
Domain 2	0.634%	0.634%	0.630%	0.632%	0.078%	0.078%	0.078%	0.077%	0.081%	0.080%	0.080%	0.079%	0.059%	0.063%	0.058%	0.055%
Domain 3	0.663%	0.664%	0.662%	0.663%	0.082%	0.082%	0.082%	0.084%	0.083%	0.083%	0.082%	0.082%	0.059%	0.064%	0.062%	0.060%
Domain 4	0.072%	0.072%	0.072%	0.071%	0.009%	0.009%	0.009%	0.009%	0.009%	0.009%	0.009%	0.009%	0.007%	0.008%	0.007%	0.007%
Domain 5	6.024%	6.037%	6.024%	6.026%	0.760%	0.754%	0.757%	0.757%	0.755%	0.751%	0.756%	0.751%	0.346%	0.343%	0.355%	0.357%
Domain 6	0.524%	0.525%	0.525%	0.523%	0.067%	0.065%	0.065%	0.064%	0.067%	0.066%	0.065%	0.067%	0.028%	0.033%	0.036%	0.034%
Domain 7	0.369%	0.369%	0.368%	0.368%	0.045%	0.045%	0.046%	0.048%	0.047%	0.047%	0.046%	0.046%	0.031%	0.036%	0.027%	0.030%
Domain 8	1.043%	1.042%	1.041%	1.042%	0.131%	0.131%	0.132%	0.131%	0.133%	0.128%	0.132%	0.132%	0.077%	0.060%	0.066%	0.066%
Domain 9	18.474%	18.462%	18.464%	18.466%	2.307%	2.300%	2.299%	2.305%	2.305%	2.310%	2.312%	2.303%	2.02%	2.63%	3.42%	2.92%
Domain 10	0.005%	0.005%	0.005%	0.005%	0.001%	0.000%	0.001%	0.001%	0.001%	0.001%	0.001%	0.001%	0.001%	0.001%	0.001%	0.001%
Domain 11	0.462%	0.465%	0.464%	0.464%	0.057%	0.058%	0.057%	0.057%	0.057%	0.059%	0.059%	0.056%	0.029%	0.034%	0.021%	0.034%
Domain 12	9.737%	9.720%	9.735%	9.730%	1.218%	1.217%	1.217%	1.216%	1.216%	1.214%	1.209%	1.216%	0.401%	0.373%	0.352%	0.377%
Domain 13	5.915%	5.916%	5.923%	5.917%	0.741%	0.740%	0.737%	0.739%	0.736%	0.735%	0.733%	0.740%	0.511%	0.497%	0.478%	0.478%
Domain 14	1.454%	1.456%	1.457%	1.455%	0.183%	0.184%	0.183%	0.184%	0.181%	0.182%	0.183%	0.182%	0.162%	0.171%	0.172%	0.155%
Domain 15	13.658%	13.667%	13.666%	13.663%	1.703%	1.717%	1.705%	1.710%	1.709%	1.710%	1.714%	1.708%	0.998%	1.101%	1.100%	1.026%
Domain 16	3.470%	3.471%	3.472%	3.474%	0.435%	0.434%	0.437%	0.432%	0.435%	0.435%	0.435%	0.439%	0.209%	0.245%	0.231%	0.234%
Domain 17	0.028%	0.029%	0.029%	0.029%	0.004%	0.004%	0.004%	0.003%	0.004%	0.003%	0.004%	0.004%	0.003%	0.003%	0.003%	0.003%
Domain 18	0.655%	0.656%	0.657%	0.659%	0.082%	0.083%	0.083%	0.083%	0.082%	0.082%	0.084%	0.081%	0.026%	0.029%	0.030%	0.021%
Domain 19	0.428%	0.429%	0.428%	0.427%	0.053%	0.053%	0.053%	0.053%	0.055%	0.053%	0.053%	0.052%	0.046%	0.046%	0.046%	0.047%
Domain 20	10.209%	10.210%	10.211%	10.214%	1.271%	1.276%	1.283%	1.277%	1.273%	1.277%	1.272%	1.279%	0.678%	0.642%	0.706%	0.923%
Domain 21	0.046%	0.045%	0.045%	0.044%	0.006%	0.006%	0.005%	0.006%	0.006%	0.006%	0.006%	0.006%	0.003%	0.003%	0.003%	0.002%
Domain 22	1.247%	1.253%	1.250%	1.252%	0.159%	0.155%	0.159%	0.156%	0.156%	0.160%	0.157%	0.158%	0.128%	0.129%	0.107%	0.120%
Domain 23	0.223%	0.223%	0.222%	0.221%	0.028%	0.029%	0.029%	0.028%	0.028%	0.028%	0.029%	0.028%	0.026%	0.026%	0.026%	0.025%
Domain 24	2.986%	2.981%	2.978%	2.984%	0.372%	0.369%	0.372%	0.370%	0.373%	0.373%	0.372%	0.375%	0.265%	0.289%	0.289%	0.270%
Domain 25	1.617%	1.614%	1.617%	1.613%	0.202%	0.203%	0.204%	0.203%	0.202%	0.202%	0.201%	0.202%	0.083%	0.091%	0.098%	0.083%
Domain 26	0.042%	0.042%	0.042%	0.042%	0.006%	0.005%	0.005%	0.005%	0.005%	0.005%	0.006%	0.005%	0.004%	0.003%	0.004%	0.004%
Total Fraction	80.000%	80.000%	80.000%	80.000%	10.000%	10.000%	10.000%	10.000%	10.000%	10.000%	10.000%	10.000%	4.386%	4.554%	4.622%	4.708%