Distilling Multilingual Transformers into CNNs for Scalable Intent Classification

Besnik Fetahu, Akash Veeragouni, Oleg Rokhlenko, and **Shervin Malmasi**
Amazon.com Inc., Seattle, WA, USA
{besnikf,avveerag,olegro,malmasi}@amazon.com

Abstract

We describe an application of Knowledge Distillation used to distill and deploy multilingual Transformer models for voice assistants, enabling text classification for customers globally. Transformers have set new state-of-the-art results for tasks like intent classification, and multilingual models exploit cross-lingual transfer to allow serving requests across 100+ languages. However, their prohibitive inference time makes them impractical to deploy in real-world scenarios with low latency requirements, such as is the case of voice assistants.

We address the problem of cross-architecture distillation of multilingual Transformers to simpler models, while maintaining multilinguality without performance degradation. Training multilingual student models has received little attention, and is our main focus.

We show that a teacher-student framework, where the teacher's unscaled activations (logits) on unlabelled data are used to supervise student model training, enables distillation of Transformers into efficient multilingual CNN models. Our student model achieves equivalent performance as the teacher, and outperforms a similar model trained on the labelled data used to train the teacher model. This approach has enabled us to accurately serve global customer requests at speed (18x improvement), scale, and low cost.

1 Introduction

For nearly all natural language understanding tasks, e.g. SuperGLUE (Wang et al., 2019), state-of-the-art results are obtained using pre-trained Transformer models. Their performance is dependent on their size and the amount of pre-training data, typically billions of tokens (Xue et al., 2021).

Intent Classification (IC), the task of understanding a user's intent from an utterance, is a core component of all voice assistants such as Siri or Alexa. IC is challenging due to the hundreds of intents and contexts that such systems must support, and IC performance has benefited greatly from Transformers (Chen et al., 2019). As voice systems have expanded support to new languages, the benefits of Transformers have multiplied with the advent of multilingual versions such as XLM-RoBERTa (Conneau et al., 2020).

Despite the advantages, deploying Transformers at scale is not always feasible, mainly due to: (i) large memory footprint (hundreds of GB),[1] and (ii) long inference time[2] that is prohibitive for applications processing millions of inputs per minute.

While approaches to reducing memory footprint — such as quantization (Vargaftik et al., 2021) or pruning (Gordon et al., 2020) — have been proposed, minimizing inference time is more challenging. Pruning can speed up inference, but there are limitations to how many self-attention layers can be pruned without loss of performance. *Knowledge Distillation* (KD) (Hinton et al., 2015) is another approach for transferring knowledge across model architectures, e.g. from Transformers to LSTMs (Wasserblat et al., 2020), to optimize performance.

However, cross-architecture distillation of multilingual Transformers to multilingual non-Transformer architectures has received almost no attention in the community. In this work we present the first exposition of this task. Specifically, we describe an approach used to deploy multilingual IC models for voice assistants allowing accurate inference at scale, speed, and low-cost.

We face two key challenges: (i) meeting *low inference* latency requirements, allowing us to globally serve customers in real time (millions per minute), and (ii) supporting *multi-linguality*, here we support 11 locales with 7 languages. Example utterances are shown below, which represent *e-commerce questions* issued in different languages.

[1] e.g. GPT-3 (Brown et al., 2020) contains 175B parameters, roughly requiring 350GB, when using `float16`.

[2] Self-attention layers have quadratic time-complexity.

Proceedings of EMNLP 2022 Industry Track, pages 439–449
December 9–11, 2020. ©2022 Association for Computational Linguistics

- how many calories are in a banana? (EN)
- wie viel fett enthält hühnchen? (DE)
- come si conservano le vongole in frigo (IT)
- cómo se hace un queque de yogur (ES)
- combien de temps peut-on réfrigérer une banane (FR)
- é possível congelar pastéis de nata (PT)

We use the teacher-student distillation paradigm, and show the optimal KD strategy for multilingual IC can leverage teacher logits alone (Mukherjee and Awadallah, 2020). Utterances for IC are typically 10-40 tokens, allowing us to exploit an efficient ConvNet architecture, and assess how they can obtain multilingual and pretrained knowledge from models like XLM-R via distillation.

While there have been previous attempts on distilling transformer models into ConvNets (Chia et al., 2019), our work is the first to explore cross-architecture multilingual KD on real-world applications with strict requirements for *latency* and *accuracy*. We make the following contributions.

- Knowledge distillation from Transformers to multilingual student (ConvNet) for intent classification based on the teacher-student paradigm;
- Minimal inference latency multilingual student models (18x speed up relative to teacher) without any loss in classification accuracy.
- Evaluation framework outlining the amount of distillation data required, and assessment of the student model's generalization on unseen data.

2 Related Work

We now review some of the popular approaches for distilling and compressing Transformer models.

Model Finetuning. Eisenschlos et al. (2019) propose an efficient way to fine tune monolingual models on multilingual tasks by simply using the output of cross-lingual Transformer models as pseudo-labels. Their approach is based on the ULMFiT model (Howard and Ruder, 2018), where instead of the stacked LSTM networks (Hochreiter and Schmidhuber, 1997), they rely on quasi recurrent neural networks (Bradbury et al.) (QRNN). QRNN are similar to CNN, with the difference that the convoluational operators are done at each timestep, however, due to parallelization, they can be computed much more efficiently than LSTMs.

QRNNs are up to 16x faster than LSTMs, however, for our case, we find that ConvNets are more efficient than QRNNs, as they do not perform step-wise computations as QRNNs do. We compare the inference time of QRNNs and our proposed student model, and conclude that simple ConvNets have significantly lower inference time.

Model Compression. Ganesh et al. (2021) systematically review approaches for compressing transformers. To reduce memory usage, quantization is often applied (Vargaftik et al., 2021). Quantization reduces the amount of bits required to store network parameters. For example, parameters represented using `float32`, can instead be stored using only 16 or fewer bits, reducing memory usage significantly. This allows deploying larger models in compute infrastructure with limited resources.

Model pruning is a widely explored research direction for compression, mainly consisting of two techniques. First, in *unstructured pruning*, weights are zeroed out using different strategies (Gordon et al., 2020). Second, in *structured pruning* either the self-attention heads (Fan et al., 2019) or the encoder layers (Hou et al., 2020) are pruned.

Quantization and pruning facilitate usage of large transformers without the requirement of very high memory capacity (GPU or CPU) machines. Quantization, and unstructured model pruning, mainly reduce memory usage. Structured pruning, where encoder and self-attention layers are dropped, can improve efficiency. Yet, for many real-world applications the latency needs cannot be met (with few milliseconds, as is our case). For instance, pruning more than 50% of attention heads can lead to performance loss (Fan et al., 2019).

Knowledge Distillation (KD). Hinton et al. (2015) discuss the trade-offs between model size and performance. Training a larger model, and distilling its knowledge to a smaller model, either using the same training data or unsupervised training data, yields identical performance. The contrary cannot be said when training a small model directly, where the performance is significantly worse than its bigger counterpart. KD works under the *teacher-student* paradigm, where the teacher's output is used to train the student model such that it mimics the teacher model in terms of the output.

There are several efforts in distilling transformers into recurrent (Wasserblat et al., 2020) and convolutional architectures (Chia et al., 2019). While recurrent models like LSTMs can significantly reduce memory footprint and latency, the step-wise sequential computation induces a large latency overhead that cannot be overcome. Conversely, ConvNets are highly efficient for text classification, both in terms of performance and latency.

Our approach is similar to that of Chia et al. (2019), in that we use CNNs as the main building block of the student model, However, we differ in several fundamental aspects and make contributions that further push the application of knowledge distillation. First, we deal with a multilingual task, which increases the complexity of the knowledge transfer from the teacher to the student model. Second, our ConvNet architecture is different to account for the multilingual requirement. Thirdly, we rely on unsupervised data for distillation, where we show how much data is necessary across different languages to have identical performance between the teacher and student models.

3 Multilingual Distillation Method

We now describe the KD approach: the IC task, the teacher/student models, and the learning objective.

3.1 IC Task

Our intent classification task requires categorizing utterances into two intents: *Commerce Question* (CQ), which are questions to the voice assistant about consumer products, and *Non-Commerce Question* (NCQ), which are all other questions.

3.2 Teacher and Student Models

Teacher Model: As our classifier is deployed globally in many languages, we use the multilingual XLM-RoBERTa (XLMR) transformer (Conneau et al., 2020) as our teacher model.

Given an utterance $\mathbf{w} = (w_1, \ldots, w_n)$, consisting of *n* tokens, the teacher model is used to encode the input, $\mathbf{T}(\mathbf{w}) = \mathbf{h}_T(\mathbf{w})$, where $\mathbf{h}_T(\mathbf{w}) \in \mathbb{R}^m$ represents the `[CLS]` pooling representation from the last XLMR layer. This is fed to a softmax classification head, consisting of a dense projection that yields the raw activations of the network (i.e. unscaled log probabilities, or logits), which are then normalized to probabilities via softmax:

$$logits_T(\mathbf{w}) = \mathbf{h}_T(\mathbf{w})^T \cdot \mathbf{W}_T \quad (1)$$

$$p_T(\mathbf{w}) = \mathtt{softmax}(logits_T(\mathbf{w})) \quad (2)$$

where $\mathbf{W} \in \mathbb{R}^{m \times C}$, C is the number of intent classes, and $logits_T(\mathbf{w})$ captures the intent of the utterance, and is used to student training.

Student Model: Figure 1 shows our student model architecture. We use a deep convolutional model (ConvNet) (LeCun and Bengio, 1995), which are widely used for text classification (Kim, 2014), mainly for two reasons. Firstly, their convolutional operators allow for effective extraction of local subword interactions in an utterance, allowing to connect question shapes (e.g. "*how many calories*") and product names (e.g. *banana*). Secondly, convolutional operations can be computed in parallel, allowing for minimal inference time, an important prerequisite for real-world applications. Finally, as IC utterances are typically short (10-40 tokens), CNNs can sufficiently capture all the important local/global lexical cues for the IC task.

Tokenization and Word Representations: Utterances are tokenized using the byte-pair encoding tokenizer model (Sennrich et al., 2016). To create a multilingual ConvNet, we leverage pretrained multilingual subword embeddings (Heinzerling and Strube, 2018). This approach allows representations of all languages, with a small vocabulary.

Encoder: Five 1D kernels of size 2-6 tokens, each with 500 filters, are aggregated with max-pooling. The pooled outputs are concatenated to form the final text representation.

Next, the student model computes the utterance representation (cf. Figure 1 (e)), $\mathbf{S}(\mathbf{w};\theta) = \mathbf{h}_S \in \mathbb{R}^m$, that is used to predict the intent probability:

$$logits_S(\mathbf{w}) = \mathbf{h}_S(\mathbf{w})^T \cdot \mathbf{W}_S \quad (3)$$

$$p_S(\mathbf{w}) = \mathtt{softmax}\,(logits_S(\mathbf{w})) \quad (4)$$

where $\mathbf{W}_S \in \mathbb{R}^{m \times C}$ and θ represent the student model parameters that need to be optimized.

3.3 Distillation Learning Objective

We use soft targets from the teacher, i.e. the unscaled log probabilities prior to softmax normalization (the logits), to train the student. We directly supervise the training of the student model $\mathbf{S}(\mathbf{w};\theta)$ such that $logits_S(\mathbf{w}) \approx logits_T(\mathbf{w})$.

To this end, our learning objective is to minimize the Mean Squared Error (MSE) loss over the logits (Mukherjee and Awadallah, 2020), computed over the N unlabelled instances:

$$\mathcal{L} = \frac{1}{N} \sum_{i=1}^{N} \|logits_S(\mathbf{w}_i) - logits_T(\mathbf{w}_i)\|^2 \quad (5)$$

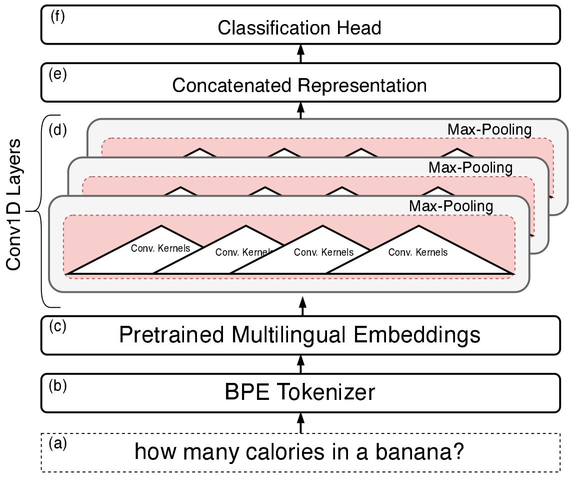

Figure 1: Student model: (a) input text, (b) byte-pair tokenizer, (c) pretrained embeddings compute subword representation; (d) 1D kernels with max-pooling; (e) concatenated representations; (f) output classification.

This logit loss encourages the student to output the same unnormalized activations as the teacher, which result in the same probabilities when normalized, and is more numerically stable to train. By minimizing $\mathcal{L}$ on a large sample of unlabelled data, the distillation process can successfully transfer the intent classification knowledge from the teacher to the student. In this aspect, it is important to consider a *large* and *representative* sample given that $\mathcal{L}$ can be minimal for a specific set of utterances, i.e. $|logits_S(\mathbf{w}) - logits_T(\mathbf{w})| < \epsilon$, however, for unseen utterances the difference between $|logits_S(\mathbf{w}) - logits_T(\mathbf{w})| \gg \epsilon$ (for some value that induces change in utterance's label.)

4 Experimental Setup

We now describe the datasets used to train the teacher model, and for distillation. We also define the evaluation metrics used to asses how well the student model mimics its teacher.

4.1 Datasets

We use 3 types of data: (i) *teacher datasets* (supervised IC training data); (ii) *student datasets*, unannotated utterances to train the student; and (iii) *test data* used to evaluate the teacher and student.

Teacher Datasets: Table 1 (a) shows details of the training data used for the teacher model. Utterances come from 7 different languages and 11 locales. The task is imbalanced, but for confidentiality, the class distribution cannot be disclosed.

		(a)	(b)	(c)
Language	Locale	Teacher instances	Distillation instances	Test instances
English	`en-US`	1.7M	4M	733k
	`en-GB`	443k	32k	280k
Spanish	`es-ES`	7.5k	2.9M	106k
	`es-US`	6.8k	1.7M	11k
	`es-MX`	6.8k	1.4M	61k
French	`fr-FR`	7.3k	2.4M	62k
	`fr-CA`	11.6k	911.3k	10.8k
German	`de-DE`	1M	3M	208k
Italian	`it-IT`	7.4k	3M	11k
Portuguese	`pt-BR`	11k	1.3M	30k
Hindi	`hi-IN`	12.6k	647k	45k
		3.5M	22.6M	1.7M

Table 1: (a) Teacher data includes 3.5M utterances with annotated binary labels. (b) Student data has 22.6M unannotated utterances used to train the student model. (c) Test instances are used to evaluate both models.

Distillation Datasets: Table 1 (b) shows the statistics of the distillation data. We randomly sample a target number of utterances from each locale over a 1-month period. The data is unlabelled. Using unsupervised data allows the KD process to transfer any of Transformer's pretrained knowledge that may not overlap with our supervised set.

Test Datasets Table 1 (c) shows the test datasets used to evaluate the performance of our teacher and student models. In total, our test set across all locales consists of 1.7M labelled instances.

4.2 Teacher and Student Configuration

Teacher Model: Model **T** is based on XLMR *base* model[3] with a total of 278M parameters, is fine-tuned on data from Table 1 for our multilingual IC task. The model is trained by minimizing the cross-entropy loss function using the AdamW (Loshchilov and Hutter, 2019) optimizer with a learning rate of $lr = 3e - 5$.

Distilled Student Model: Model **S** is described in Figure 1, and consists of a total of 103M parameters. It is trained using the data in Table 1 (b) by minimizing the loss in Equation (5). A dropout rate of 10% is applied to the embeddings and CNN filters for regularization. We fine-tune the pretrained embeddings, and apply learning rate warmup over the first 2 epochs to prevent catastrophic forgetting. We train for 50 epochs (via the Adam optimizer), with an early stopping criterion of 3 consecutive epochs of non-decreasing loss.

[3] `https://huggingface.co/xlm-roberta-base`

4.3 Baseline

Our main objective is minimizing inference latency of Transformer models for IC. IC accuracy is not problematic for in-domain data, and most models achieve high performance (Larson et al., 2019).

QRNN: We focus in comparing only w.r.t the inference time between different approaches.[4] We compare **S** to QRNN, proposed in (Bradbury et al.), and consider two configurations: (i) $\textbf{QRNN}_4$: with 4 ConvNet layers (as reported in Bradbury et al.), and (ii) $\textbf{QRNN}_5$: with 5 ConvNet layers, equivalent to the layers used in **S**.

Supervised Student Model: To assess whether distillation of teacher's knowledge into **S** using unlabelled data is needed in the cases of abundance of labelled training data, we additionally train an identical model to **S** using the supervised training data in Table 1 instead, which we denote with $\mathbf{S}_{sup}$. The training loss for $\mathbf{S}_{sup}$ is the cross-entropy loss.

4.4 Evaluation Metrics

Accuracy: We measure performance based on Precision (P) and Recall (R). Specifically, we compare the models at the threshold-agnostic P/R Break-Even Point (PR-BEP) (Joachims, 2005), where the precision and recall of the model are equal. To compare performance over all thresholds, we report PR-AUC (area under the PR Curve) which is a meaningful metric for imbalanced tasks (Liu et al., 2019). Due to confidentiality, we report only the gap of **S** and $\mathbf{S}_{sup}$ to **T**, as their absolute difference.

Efficiency: We measure wall-time t to compute the inference latency in *milliseconds* (ms). All measurements are the averaged latency over 100 trials, computed on an `m5.4xlarge` instance.[5]

5 Results

5.1 Model Accuracy

Overall Performance: Table 2 shows the performance difference between the teacher and student models. The overall PR-BEP gap across locales with 0.1% between **T** and **S** is negligible.[6]

Contrary to **S**, $\mathbf{S}_{sup}$ has a large gap to **T**, with an overall difference of 6%, and in certain locales, exceeding 30% in terms of PR-AUC. This gap remains large in the languages with the largest amount of supervised data, but is much more prominent in those with little data. This highlights two main findings: (i) **T** due to its Transformer architecture has a superior learning capacity when compared to directly training $\mathbf{S}_{sup}$ in a supervised manner; (ii) knowledge distillation allows us to successfully transfer the teacher's pretrained knowledge to the student, allowing the student to acquire knowledge not present in the labeled data, and achieve similar generalization as the teacher (cf. Appendix A).

	S		$\mathbf{S}_{sup}$	
	PR AUC	PR BEP	PR AUC	PR BEP
`en-US`	▼ 0.2%	▼ 0.2%	▼ 6%	▼ 6%
`en-GB`	▲ 0.2%	▲ 0.8%	▼ 5%	▼ 3%
`es-ES`	▲ 0.2%	▼ 0.1%	▼ 8%	▼ 6%
`es-US`	▼ 1.3%	▼ 0.9%	▼ 8%	▼ 7%
`es-MX`	▼ 0.4%	▼ 0.4%	▼ 10%	▼ 7%
`fr-FR`	▲ 0.1%	▲ 0.1%	▼ 8%	▼ 6%
`fr-CA`	▲ 0.4%	▲ 0.8%	▼ 7%	▼ 6%
`de-DE`	▼ 0.4%	▼ 0.6%	▼ 6%	▼ 5%
`it-IT`	▼ 0.3%	▼ 1.0%	▼ 11%	▼ 9%
`pt-BR`	▲ 0.2%	▼ 0.1%	▼ 33%	▼ 25%
`hi-IN`	▲ 0.2%	▼ 1.0%	▼ 30%	▼ 24%
average	▼ 0.1%	▼ 0.2%	▼ 6%	▼ 6%

Table 2: The gap between the student models (**S** and $\mathbf{S}_{sup}$) to **T** is reported for the same test set. For **S** the gap of 0.2% is marginal, whereas for $\mathbf{S}_{sup}$ the gaps are highly significant according to the Binomial proportion test (*p–value* < 0.01).

Incremental Distillation Performance: Table 3 shows the gap in performance between the student and teacher models, for varying amount of data used to train the student model. The data from Table 1 (b) is sampled using stratified sampling, with the locales representing the groups.

With only 1% of the data, the gap in terms of PR-BEP is 8.7% absolute points. Increasing this to 10% or more, the gap closes to less than 1%. Concretely, 10% represents 2.2M instances across all locales. In real-world settings it is reasonably cheap to obtain such amounts of unlabelled data.

Results indicate that with appropriate data, logit loss is highly effective for capturing the teacher's knowledge. The student, using a different tokenizer and subword embeddings, is able to match teacher performance. Relative to other methods, logit loss is simpler to implement, and faster to train. For the IC task, we did not need to distill internal model

[4] Experimental evaluation shows that **S** achieves nearly identical performance to **T**. Thus, we do not report accuracy metrics for QRNN, given that its inference latency is higher.

[5] `https://aws.amazon.com/ec2/instance-types/`

[6] Statistically not significant per Binomial Proportion Test.

PR-BEP Absolute Percentage Difference to the Teacher Model.											
	1%	5%	10%	20%	30%	40%	50%	60%	70%	80%	90%
en-US	▼ 8.60%	▼ 2.70%	▼ 1.40%	▼ 0.90%	▼ 0.70%	▼ 0.30%	▼ 0.40%	▼ 0.20%	▼ 0.40%	▼ 0.60%	▼ 0.20%
en-GB	▼ 7.30%	▼ 2.10%	▼ 0.80%	▼ 0.50%	▼ 0.70%	▼ 0.10%	▼ 0.30%	0%	▲ 0.40%	▲ 0.20%	▲ 0.10%
es-ES	▼ 9.40%	▼ 4.40%	▼ 1.40%	▼ 0.70%	▼ 0.60%	▼ 0.80%	▼ 0.80%	▼ 0.90%	▼ 0.50%	▼ 0.70%	▼ 1.00%
es-US	▼ 9.00%	▼ 3.90%	▼ 3.60%	▼ 0.90%	▼ 0.60%	▼ 1.20%	▲ 0.60%	▲ 0.30%	▲ 0.90%	▲ 0.90%	▼ 1.20%
es-MX	▼ 9.80%	▼ 4.60%	▼ 1.90%	▼ 1.50%	▼ 0.80%	▼ 1.40%	▼ 0.80%	▼ 0.60%	▼ 0.40%	▼ 0.90%	▼ 1.20%
fr-FR	▼ 8.30%	▼ 3.10%	▼ 1.30%	▼ 1.60%	▼ 0.90%	▼ 1.10%	▼ 0.80%	▼ 0.70%	▼ 0.80%	▼ 1.00%	▼ 0.70%
fr-CA	▼ 9.50%	▼ 3.00%	▼ 0.30%	▼ 0.30%	▼ 0.30%	▲ 0.30%	▲ 1.20%	▼ 0.30%	▲ 0.60%	▼ 1.20%	▼ 0.30%
de-DE	▼ 8.20%	▼ 3.30%	▼ 1.70%	▼ 1.20%	▼ 1.30%	▼ 0.70%	▼ 0.60%	▼ 0.60%	▼ 0.60%	▼ 0.50%	▼ 0.20%
it-IT	▼ 11.20%	▼ 4.30%	▼ 2.00%	▼ 1.10%	▼ 1.90%	▼ 1.20%	▼ 0.70%	▼ 1.00%	▼ 1.10%	▼ 1.10%	▼ 1.40%
pt-BR	▼ 11.80%	▼ 5.10%	▼ 1.80%	▼ 1.70%	▼ 2.60%	▼ 1.00%	▼ 0.40%	▼ 0.80%	▼ 1.40%	▼ 0.60%	▼ 1.50%
hi-IN	▼ 11.90%	▼ 8.30%	▼ 5.90%	▼ 4.30%	▼ 3.10%	▼ 2.40%	▼ 1.80%	▼ 1.40%	▼ 1.20%	▼ 1.10%	▼ 1.30%
average	▼ 9.54%	▼ 4.1%	▼ 2.01%	▼ 1.34%	▼ 1.22%	▼ 0.95%	▼ 0.76%	▼ 0.62%	▼ 0.75%	▼ 0.8%	▼ 0.82%

Table 3: PR-BEP performance of the student model trained on varying portion of the distillation data from Table 1 (b). Overall, with 1% of the data used for distillation, the student model has an average gap in terms of PR-BEP of 8.7%. With increasing percentage of data used for distillation the gap is shrunk to 0.6% for 40% of the data.

values (e.g. representation loss). We did not use the supervised data for student training (e.g. with cross-entropy loss); our finding is that a sufficiently large and representative unsupervised sample will contain samples similar to those in the supervised set, as well as dissimilar ones, thus allowing the transfer of knowledge represented by both the labelled data and the Transformer's pretrained knowledge.

5.2 Inference Latency

A drawback in deploying transformers is their prohibitive inference latency, mainly impacted by: (i) model size, and (ii) number of encoder layers.

Figure 2 shows the latency for different ablations of $\mathbf{T}$ (with varying numbers of encoder layers), and the latency of $\mathbf{S}$, as the model with the lowest latency. Comparing $\mathbf{T}$ and $\mathbf{S}$, our student model has nearly 18x lower inference latency, with only 2.7ms. This represents a drastic latency reduction, allowing us to process inputs extremely quickly.

For the teacher ablations, even for $\mathbf{T}_1$, the inference latency is still higher than that of $\mathbf{S}$, with an additional +1.24ms latency per utterance. Furthermore, pruning layers is not lossless in terms of performance, especially in this case where only one layer is retained (Fan et al., 2019). The bottom part of Figure 2 shows the gap of the different pruned teacher models $\mathbf{T}_l$ w.r.t the full model $\mathbf{T}$. The gap is high when we use fewer than 8 layers, with more than 12% drop in PR-BEP. It is clear that there is no clear trade-off between self-attention layer pruning and inference latency reduction.

Finally, comparing the baseline $\mathbf{QRNN}_4$ and $\mathbf{QRNN}_5$, we note that the proposed student architecture, relying solely on ConvNets, results in a significantly lower inference latency. Our student architecture has 3.8x and 2.95x lower latency than $\mathbf{QRNN}_5$ and $\mathbf{QRNN}_4$, respectively. This significant increase in terms of latency can be explained by the fact that $\mathbf{QRNN}$ applies its convolutional operators for timestep (each token in an utterance), which although more efficient than LSTMs (due to parallelization), it introduces a significant overhead over the traditional ConvNet architectures.

6 Conclusions

We described an approach for distilling knowledge from Transformer into a single multilingual CNN. To our knowledge this is the first detailed exposition of cross-architecture KD to multilingual student models. We leverage the outlined framework to accurately serve predictions for our customers at speed, scale, low-cost, and across all languages. Empirically we showed how such a KD framework can be utilized in practice:

1. With sufficient unsupervised data, leveraging logits is an optimal distillation strategy for training smaller and more efficient student models, without significant performance loss.

2. KD allows smaller and more efficient models to mimic the performance of their teacher counterparts, which is not the case if similar architectures are directly trained using labelled data.

3. KD is highly preferred over other techniques such as pruning. Transformers, even with a single encoder layer have higher inference latency, and the performance drop with pruning is large, where $\mathbf{T}_1$ has a 23% and 22% gap in terms of PR-BEP w.r.t $\mathbf{T}$ and $\mathbf{S}$, respectively.

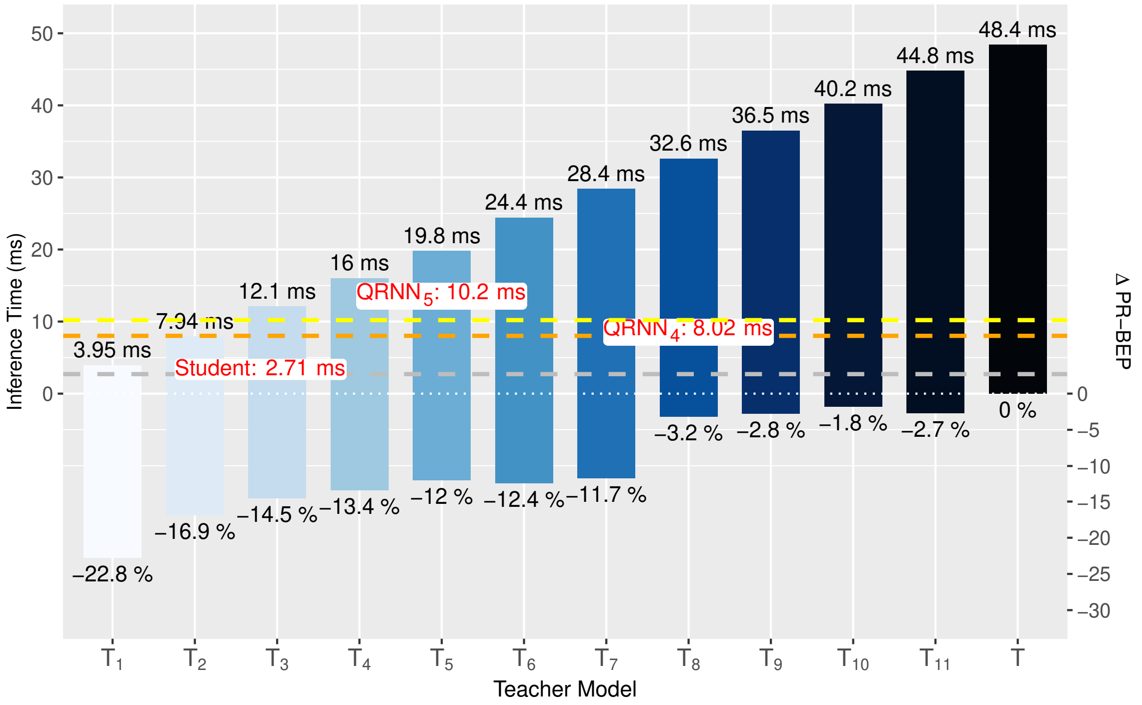

Figure 2: The upper plot shows the *inference latency* (in milliseconds) for the teacher models, where $\mathbf{T}_l$ ($l \in \{1, \ldots, 11\}$). $\mathbf{T}_1$ is a single encoder layer, with the other 11 layers pruned. The bottom plot shows the gap in terms of PR-BEP of the $\mathbf{T}_l$ models to the full teacher model. Note that, $\mathbf{T}_1$ which has the closest inference time to $\mathbf{S}$ (with 2.71ms latency), has a 22.8% and 22.5% drop in terms of PR-BEP w.r.t $\mathbf{T}$ and $\mathbf{S}$, respectively. Similarly, for $\mathbf{QRNN}$, the inference time is shown in the orange and yellow dashed lines, with a latency of $\mathbf{QRNN}_4 = 8.02$ *ms* and $\mathbf{QRNN}_5 = 10.02$ *ms*.

4. For IC, a single multilingual CNN using multilingual subword embeddings can match the teacher performance despite using a different tokenizer. It is highly efficient, decreasing the latency by nearly 18x relative to the teacher.

5. Using as few as 2-3M distillation instances, **S** achieves highly comparable performance as **T**, with less than 1% PR-BEP difference. The gap diminishes to 0.95% with just 40% of distillation data, specifically 8.8M instances.

6. **S** achieves the same generalization power as **T**. On a held out test set (unseen during training and distillation), the output probabilities have a very low KL divergence (cf. Appendix A).

References

James Bradbury, Stephen Merity, Caiming Xiong, and Richard Socher. Quasi-recurrent neural networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared 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 M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher 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 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Qian Chen, Zhu Zhuo, and Wen Wang. 2019. Bert for joint intent classification and slot filling. *arXiv preprint arXiv:1902.10909*.

Yew Ken Chia, Sam Witteveen, and Martin Andrews. 2019. Transformer to CNN: label-scarce distillation for efficient text classification. *CoRR*, abs/1909.03508.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In

Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020, pages 8440–8451. Association for Computational Linguistics.

Susan Dean and Barbara Illowsky. 2018. Descriptive statistics: skewness and the mean, median, and mode. *Connexions website*.

Julian Eisenschlos, Sebastian Ruder, Piotr Czapla, Marcin Kardas, Sylvain Gugger, and Jeremy Howard. 2019. Multifit: Efficient multi-lingual language model fine-tuning. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 5701–5706. Association for Computational Linguistics.

Angela Fan, Edouard Grave, and Armand Joulin. 2019. Reducing transformer depth on demand with structured dropout. *arXiv preprint arXiv:1909.11556*.

Prakhar Ganesh, Yao Chen, Xin Lou, Mohammad Ali Khan, Yin Yang, Hassan Sajjad, Preslav Nakov, Deming Chen, and Marianne Winslett. 2021. Compressing Large-Scale Transformer-Based Models: A Case Study on BERT. *Transactions of the Association for Computational Linguistics*, 9:1061–1080.

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*.

Benjamin Heinzerling and Michael Strube. 2018. Bpemb: Tokenization-free pre-trained subword embeddings in 275 languages. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation, LREC 2018, Miyazaki, Japan, May 7-12, 2018*. European Language Resources Association (ELRA).

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.

Lu Hou, Zhiqi Huang, Lifeng Shang, Xin Jiang, Xiao Chen, and Qun Liu. 2020. Dynabert: Dynamic BERT with adaptive width and depth. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 328–339, Melbourne, Australia. Association for Computational Linguistics.

Thorsten Joachims. 2005. A support vector method for multivariate performance measures. In *Proceedings of the 22nd international conference on Machine learning*, pages 377–384.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751, Doha, Qatar. Association for Computational Linguistics.

Stefan Larson, Anish Mahendran, Joseph J. Peper, Christopher Clarke, Andrew Lee, Parker Hill, Jonathan K. Kummerfeld, Kevin Leach, Michael A. Laurenzano, Lingjia Tang, and Jason Mars. 2019. An evaluation dataset for intent classification and out-of-scope prediction. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 1311–1316. Association for Computational Linguistics.

Yann LeCun and Yoshua Bengio. 1995. Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*, 3361(10):1995.

Shengchao Liu, Yingyu Liang, and Anthony Gitter. 2019. Loss-balanced task weighting to reduce negative transfer in multi-task learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 9977–9978.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Subhabrata Mukherjee and Ahmed Hassan Awadallah. 2020. Xtremedistil: Multi-stage distillation for massive multilingual models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 2221–2234. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*. The Association for Computer Linguistics.

Shay Vargaftik, Ran Ben Basat, Amit Portnoy, Gal Mendelson, Yaniv Ben-Itzhak, and Michael Mitzenmacher. 2021. Communication-efficient federated learning via robust distributed mean estimation. *CoRR*, abs/2108.08842.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. Superglue: A stickier benchmark for general-purpose language understanding systems. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 3261–3275.

Moshe Wasserblat, Oren Pereg, and Peter Izsak. 2020. Exploring the boundaries of low-resource bert distillation. In *Proceedings of SustaiNLP: Workshop on Simple and Efficient Natural Language Processing*, pages 35–40.

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, NAACL-HLT 2021, Online, June 6-11, 2021*, pages 483–498. Association for Computational Linguistics.

Appendix

A Generalizability of Distillation

Distribution Loyalty: We now assess the generalization of the KD process: whether the teacher and student behave similarly on unseen data. From the test set in Table 1, we take the instances that are not present in either the teacher or distillation data, resulting in 806k instances. We measure the Kullback-Leibler (KL) divergence between the output probabilities of the teacher and student models (trained with varying number of instances). KL values closer to zero, reflect similar distributions and behavior of the two models on the unseen data.

Figure 3 shows that with increasing amount of distillation data, the teacher and student models output highly similar probabilities. Further, we note that in some cases, such as for `hi-IN`, with 1% of the distillation data (or 6472 instances), the two models output highly diverging probabilities. Increasing the distillation data to 30% and upwards, we see that the probability distributions become highly similar, a fact also reflected in Table 3, where the student models have very close scores.

NCQ/CQ Class Separation: Figure 4 shows the class separation for the different student models, distilled with varying amount of data (1%, 10%, and 50%), and as well as the teacher model (representing the target performance for **S**). An ideal classifier would output 0 probability for NCQ class, and 1.0 for the CQ class, given that the IC models are trained on the binary case to predict whether an utterance is a commercial question or not.

From Figure 4, it can be noted that for $\mathbf{S}_{1\%}$, there is a large amount of CQ utterances that have low CQ probability (x-axis). As the amount of distillation data is increased, we note that the distribution become more skewed (Dean and Illowsky, 2018), which represents an increase in classifier accuracy. For instance, from a skewedness score of $G1_{CQ} = -2.29$ for $\mathbf{S}_{1\%}$, we obtain a skewedness score of $G1_{CQ} = -3.71$ for $\mathbf{S}_{50\%}$, which implies that the CQ probability distribution is more skewed towards the higher CQ scores. In the case of classification models, the more skewed the distributions, the higher the classification performance.

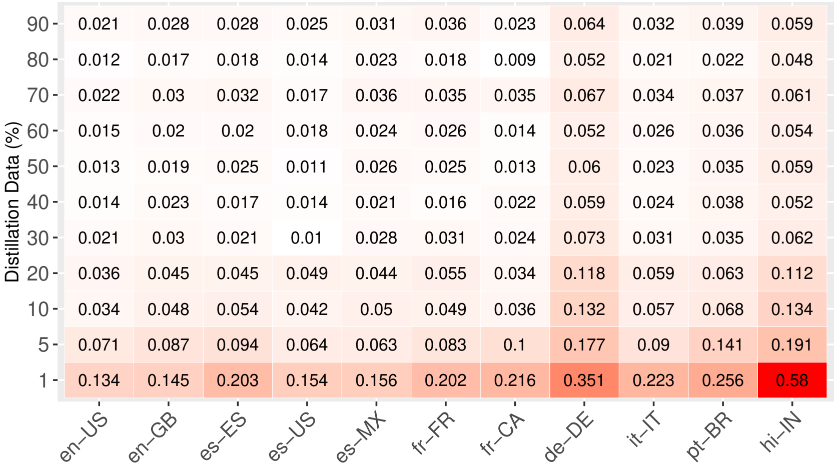

Figure 3: KL-Divergence of the confidence score distribution between teacher and student models on unseen data. With increasing amounts of distillation data, the probability distributions become highly similar.

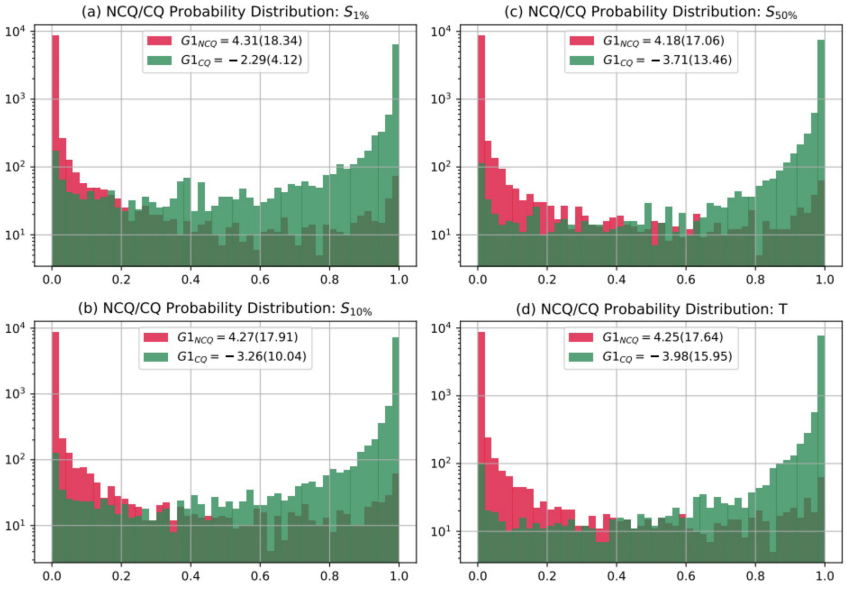

Figure 4: NCQ/CQ confidence distribution (x–axis) as a function of how likely an utterance is to be CQ. For NCQ, this probability ideally should be close to zero, and vice-versa for CQ (close to one). The skewedness score $G1$ measures the concentration of the probability mass for NCQ and CQ, respectively. For NCQ the higher score the better (in the positive range), whereas for CQ, the lower the score the better (negative range). The results are shown for the student models: $\mathbf{S_{1\%}}$, $\mathbf{S_{10\%}}$, $\mathbf{S_{50\%}}$ (distilled with 1%, 10%, and 50% of the data, respectively), and for the teacher model $\mathbf{T}$.