Legal-Tech Open Diaries: Lesson learned on how to develop and deploy light-weight models in the era of humongous Language Models

Stelios Maroudas[* †◇]   **Sotiris Legkas**[* †◇]
Prodromos Malakasiotis[†]   **Ilias Chalkidis**[‡◇]
[†] Department of Informatics, Athens University of Economics and Business, Greece
[‡] Department of Computer Science, University of Copenhagen, Denmark
[◇] Cognitiv+, Athens, Greece

Abstract

In the era of billion-parameter-sized Language Models (LMs), start-ups have to follow trends and adapt their technology accordingly. Nonetheless, there are open challenges since the development and deployment of large models comes with a need for high computational resources and has economical consequences. In this work, we follow the steps of the R&D group of a modern legal-tech start-up and present important insights on model development and deployment. We start from ground zero by pre-training multiple domain-specific multi-lingual LMs which are a better fit to contractual and regulatory text compared to the available alternatives (XLM-R). We present benchmark results of such models in a half-public half-private legal benchmark comprising 5 downstream tasks showing the impact of larger model size. Lastly, we examine the impact of a full-scale pipeline for model compression which includes: a) Parameter Pruning, b) Knowledge Distillation, and c) Quantization: The resulting models are much more efficient without sacrificing performance at large.

1 Introduction

Transformer-based Languages Models (LMs) (Radford and Narasimhan, 2018; Devlin et al., 2019; Liu et al., 2019) have stormed NLP benchmarks with state-of-the-art performance, while recently humongous billion-parameter-sized models (Brown et al., 2020; Rae et al., 2021; Hoffmann et al., 2022) have showcased impressive few-shot capabilities. In addition, multi-lingual LMs (Conneau et al., 2020) have been also developed demonstrating exceptional results as well as impressive performance in zero-shot cross-lingual transfer.

The legal NLP literature is also flourishing with the release of many new resources, including large legal corpora (Henderson* et al., 2022), benchmark datasets (Chalkidis et al., 2021a; Koreeda and Manning, 2021; Zheng et al., 2021; Chalkidis et al., 2022; Habernal et al., 2022), and pre-trained legal-oriented language models (Chalkidis et al., 2020; Zheng et al., 2021). Despite this impressive progress, the efficacy of differently-sized language models on legal NLP tasks and the importance of domain (legal) specificity are still understudied, while the effect of model compression techniques in model's performance and efficiency is ignored.

In this work, we aim to shed light in all these directions following model development across three incremental steps in a *pipelined* approach:

(a) *model pre-training* on large legal corpora,

(b) *model fine-tuning* on down-stream tasks, and

(c) *model compression* to improve efficiency.

To do so, we initially develop 4 multi-lingual legal-oriented language models (C-XLMs). We benchmark their performance across 5 down-stream legal NLP tasks, comprising both publicly available and private datasets, covering both English and multi-lingual scenarios in several tasks types, i.e., document/sentence classification, natural language inference, and entity extraction. Finally, we experiment with a full-scale pipeline for model compression which includes a) Parameter Pruning, b) Knowledge Distillation, and c) Quantization to produce much more efficient (smaller and faster) models that can be effectively deployed in production.

Our work aims to provide guidelines to legal-tech practitioners on model development (pre-training, fine-tuning, compression) bearing both performance and efficiency into consideration. Our findings show that the impact of larger vs. smaller models, domain-specific vs. generic models and the efficacy of model compression techniques varies across tasks, but in general larger domain-specific models perform better. Via full-scale model compression, we produce models with performance decrease by 2.3 p.p., while being approx.

Model Alias		#Langs	#Layers	#Units	#Heads	#Params	Vocab. Size	Train. Tokens	MLM Acc.
XLM-R	base	100	12	768	12	278M	250k	6.3T	74.0
XLM-R	large	100	24	1024	16	559M	250k	6.3T	78.9
C-XLM	tiny	10	4	128	4	9M	64k	92B	54.9
C-XLM	small	10	6	256	4	21M	64k	92B	68.9
C-XLM	base	10	12	512	8	71M	64k	92B	77.8
C-XLM	large	10	24	1024	16	368M	64k	92B	81.5

Table 1: Model Specifications, Training Tokens processed on pre-training and MLM performance (Acc.) for all variants of our XLM (C-XLM) models and the XLM-R models of Conneau et al. (2020) considered as baselines.

42× smaller, and approx. 66× faster. We also find that fully compressed models outperform equally sized distilled or fine-tuned models.

2 Model Specifications

Following Chalkidis et al. (2020), we pre-train from scratch legal domain specific transformer-based language models. Our models are based on the RoBERTa architecture (Liu et al., 2019), i.e., trained with the Masked Language Modelling (MLM) objective, excluding the Next Sentence Prediction (NSP) one used by BERT (Devlin et al., 2019). In addition, based on the industry needs and driven by the work of (Conneau et al., 2020), our models are a multilingual one -usually referred as XLM in the literature- and supports ten languages in total (English, French, German, Greek, Spanish, Italian, Dutch, Polish, Portuguese, Russian).

We pre-train 4 variants of custom XLM models (C-XLM) starting from a large version with 24 Transformer blocks (layers), each consisting of 1024 hidden units and 16 attention heads and continue by decreasing each time by a factor of 2 across all dimensions, i.e., blocks/layers, hidden units, and attention heads (Table 1).[1]

3 Pre-Training

3.1 Training Corpora

We pre-trained our models using multi-lingual corpora that consist of regulations and contracts. For regulations, we used the MultiEURLEX dataset of Chalkidis et al. (2021b) that comprises 65k EU regulations officially translated in 24 languages.[2]. We also considered additional publicly available English resources; specifically the 250 US code books, part of the "Pile of Law" corpus released by

(Henderson* et al., 2022), along-size 36k UK laws published by Chalkidis and Søgaard (2022).

Regarding contracts, we considered the LEDGAR (Tuggener et al., 2020) dataset comprising 900k sections from US contracts in English; and 60k additional full contracts in English from a publicly available crawl from EDGAR. Since, there are no publicly available contracts in the rest of the languages, we translated these documents using state-of-the-art Neural Machine Translation (NMT) systems across all languages of interest.[3]

3.2 Custom Vocabulary

Relying on the above mentioned resources, we built a custom vocabulary of 64k sub-word units that better fit the documents in the respective domains and languages of interest. We opted for Byte-Pair Encodings (BPEs) (Sennrich et al., 2016), similarly to most recent work on Transformer-based language models (Radford and Narasimhan, 2018; Liu et al., 2019; Conneau et al., 2020).

3.3 Masked Language Modelling (MLM)

We pre-trained all variants of C-XLM (our domain-specific multi-lingual RoBERTa) for 1.1m steps (gradient updates) in total based on a two-step approach, similarly to Devlin et al. (2019), i.e., pre-train for 1m steps with sequences up to 128 sub-word units, followed by continued pre-training for 100k steps with sequences up to 512 sub-word units, always with a batch size of 512 sequences.[4] At each example, we mask out 15% of the tokens in total. We train all models for a maximum learning rate of 1e−4 with warm-up for the initial (5%) training steps followed by a cosine decay.

In comparison XLM-R models were pre-trained for 1.5m steps with batches of 8192 sequences, which accounts for approx. 63× more training

---

[1]A minor exception in the tiny version, where we consider 4 attention heads of 32 hidden units per head instead of 2 attention heads with 64 units per head.

[2]In our work, we consider 9 languages (English, French, German, Greek, Spanish, Italian, Dutch, Polish, Portuguese).

[3]We used the OpusMT (en2m) mBART models using the EasyNMT library.

[4]This approach aims to a more efficient (compute-friendly) pre-training, since pre-training with shorter sequences severely decreases the needed compute and time.
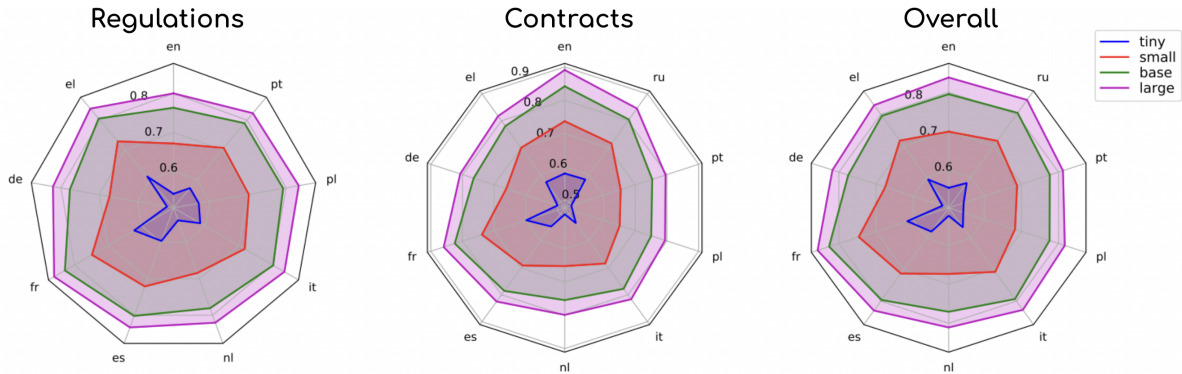
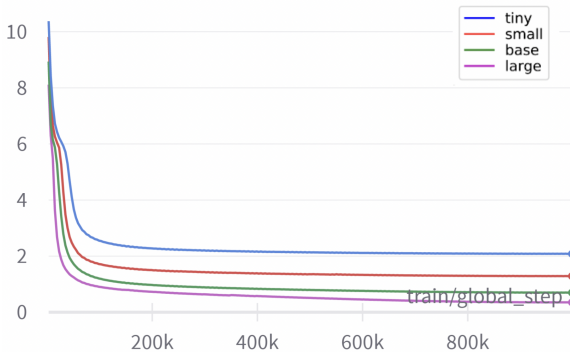Figure 1: MLM performance per language across C-XLM model variants depicted with different coloured webs.

Figure 2: Pre-training loss curves of C-XLMs.

tokens processed; the majority of those in high-resource languages like the ones we consider.

3.4 MLM Results

In Figure 2, we observe the loss curves of differently sized models during pre-training. While models are equally poor performing in the very initial steps, larger models substantially outperform the smaller counterparts due to their increased capacity (number of parameters). Table 1 presents the accuracy of our different models. As expected, the large version (81.5% accuracy) followed by the base version (77.8% accuracy) of C-XLM outperform their corresponding generic XLM-R models by 2.6% and 3.8% respectively.[5] Figure 1 presents masked language modelling performance in finer details across languages per model, highlighting the predominance of our two largest models.[6]

4 Fine-tuning

4.1 Benchmark - Tasks and Datasets

In this section, we briefly present the evaluation benchmark that we use, which consist of both publicly available and private datasets. The benchmark is diverse covering three task types (document, sentence, and token classification) and two multi-lingual datasets.[7] The datasets in detail are:

MultiEURLEX (Chalkidis et al., 2021a), a multilingual dataset for legal topic classification comprising 65k EU laws officially translated in 23 EU languages.[8] Each document (EU law) was originally annotated with relevant EUROVOC[9] concepts by the Publications Office of EU. We use the 21 'Level 1' labels, obtained by Chalkidis et al. (2021a) from the original EUROVOC annotations of the documents. We use a derivative of the original dataset considering only 1k non-parallel documents per supported language (9k in total, Section 3.1).[10] This is a multi-label document classification task, thus we evaluate performance using macro- ($m\text{-}F_1$) and micro- ($\mu\text{-}F_1$) F1 scores.

UNFAIR-ToS (Drawzeski et al., 2021) is a dataset for detecting unfair clauses in Terms of Service (ToS) agreements from on-line platforms (e.g., YouTube, Facebook, etc.) in 4 languages (English, German, Italian, and Polish). The dataset has been annotated on the sentence-level with 8 types of *un-*

---

[5]A comparison between the XLM-R models of Conneau et al. (2020) and our models (C-XLMs) is not ideal due to the different vocabulary used. Nevertheless, it provides a general idea on pre-training performance on legal specific corpora.

[6]More fine-grained MLM evaluation (per language and per document type) can be found in Appendix B.

[7]We do not use the LexGLUE benchmark of Chalkidis et al. (2022), since it is monolingual (English only) and also covers tasks that involve litigation, which are out of scope.

[8]MultiEURLEX is available at `https://huggingface.co/datasets/multi_eurlex`.

[9]EUROVOC is a hierarchically organized taxonomy of concepts (a hierarchy of labels) available at `http://eurovoc.europa.eu/`.

[10]This is inline with the work of Xenouleas et al. (2022), where the authors consider a more "realistic" harder version of MultiEURLEX with less and non-parallel documents.

Model	Alias	MultiEURLEX		UNFAIR-ToS		CNLI		Obligations		ContractNER	
		μ-F$_1$	m-F$_1$	Acc.	MAE	μ-F$_1$	m-F$_1$	μ-F$_1$	m-F$_1$	μ-F$_1$	m-F$_1$
XLM-R	(base)	75.3	53.2	86.6	0.17	84.0	81.9	89.7	88.2	92.4	93.9
XLM-R	(large)	77.8	63.8	89.0	0.16	**86.3**	**84.7**	88.9	87.4	92.8	93.7
C-XLM	(tiny)	66.5	46.1	78.2	0.27	70.2	69.2	88.7	87.4	87.2	89.3
C-XLM	(small)	72.3	54.7	85.4	0.20	79.7	77.0	90.4	89.0	90.1	92.4
C-XLM	(base)	75.3	59.4	87.3	0.18	84.0	82.1	91.2	90.4	92.9	93.9
C-XLM	(large)	**78.4**	**65.4**	**89.7**	**0.14**	85.3	83.0	**91.8**	**90.6**	**93.2**	**94.6**

Table 2: Overall results of fine-tuned models across all down-stream tasks.

fair contractual terms, meaning terms (sentences) that potentially violate user rights according to EU consumer law. Sentences have been also annotated according to a 3-level fairness score (*fair*, *partially unfair*, *clearly unfair*). In our case, we examine the latter task as sentence regression and evaluate performance using Mean Absolute Error (MAE), and Accuracy (Acc.) on rounded (discrete) scores.

ContractNLI (Koreeda and Manning, 2021) is a dataset for contract-based Natural Language Inference (NLI). The dataset consists of 607 contracts, specifically Non-Disclosure Agreements (NDAs). Each document has been paired with 17 templated *hypotheses* and labeled with one out of three classes (*entailment*, *contradiction*, or *neutral*). We examine a lenient version of this task, where instead of the full document (NDA), we represent the document with a short number of sentences which have been annotated as rationales for the specific task. This is a single-label multi-class document classification task and we evaluate performance using macro- (m-F$_1$) and micro- (μ-F$_1$) F1 scores.

Contract-Obligations (Chalkidis et al., 2018) is a proprietary (privately developed) dataset for obligation extraction from contracts (legal agreements). The dataset consists of 100 service agreements. Each contract has been split into paragraphs (approx. 9,400 in total), and labeled with 4 obligation sub-types, i.e., *Obligation*, *Deliverable*, *Discretion*, and *Prohibition*, while some paragraphs are not relevant, resulting in a total of 5 potential classes. This is a single-label multi-class document classification task. We evaluate performance using macro- (m-F$_1$) and micro- (μ-F$_1$) F1 scores.

ContractNER (Chalkidis et al., 2017) is a proprietary dataset for contract element extraction. The dataset consists of 3,500 contractual introductions from several types (service, employment, purchase, etc.) of contracts. Each introduction (paragraph)

has been labeled with 4 entity types (*Title*, *Contracting Party*, *Start Date*, *Effective Date*). This is a single-label multi-class token classification task. Thus, we evaluate performance using macro- (m-F$_1$) and micro- (μ-F$_1$) F1 scores on entity level.

4.2 Experimental Set Up

We tune all models conducting a grid search for learning rates ∈ {1e-4, 3e-4, 1e-5, 3e-5, 5e-5, 1e-6}. We use early stopping based on validation loss; we select and report test scores based on the model with the best validation performance.[11]

4.3 Fine-tuning Results

Table 2 presents the results of the fined-tuned baselines, XLM-R models, (upper zone) and of all the variants of our C-XLM models (lower zone) for each downstream task. We hypothesize that the base and large versions of C-XLM will perform better compared to their counterpart XLM-R models. Indeed, the base version of C-XLM always outperforms XLM-R across all 5 datasets, while the large version of C-XLM outperforms XLM-R in all but one (4 out of 5) datasets.

MultiEURLEX: Both large versions of C-XLM and XLM-R clearly outperform the rest of the models with the C-XLM outperforming XLM-R by 0.6 p.p. in μ-F$_1$ and 1.6 p.p. in m-F$_1$. Similarly, the base version of C-XLM outperforms the equivalent version of XLM-R. Interestingly, the small version of C-XLM has comparable performance with the latter while being approx. 13× smaller.

UNFAIR-ToS: Both large and base versions of C-XLM outperform their counterpart XLM-R models by 0.7 p.p. in accuracy. Again, the small version of C-XLM achieves competitive performance to base-sized models.

---

[11] Additional details and development scores are provided in Appendix A
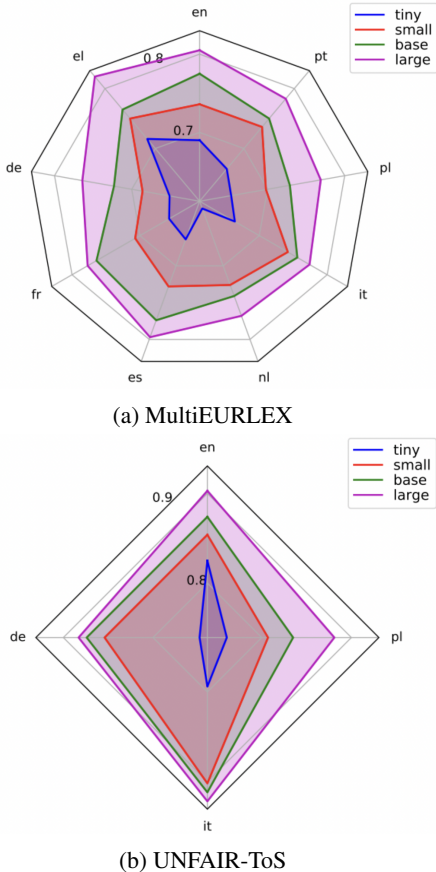
(a) MultiEURLEX

(b) UNFAIR-ToS

Figure 3: Radar plots with per language performance for the multilingual MultiEURLEX and Unfair-ToS datasets for all the versions of C-XLM.

ContractNLI: In this task, we find that the large version of XLM-R outperforms the one of C-XLM (+1 p.p. in μ-F_1 and +1.7 p.p. in m-F_1) while both base models perform comparably. We also note that the relative differences between differently sized models are the more intense across all tasks.

Contract-Obligations: On this task, all C-XLM models except the tiny version outperform the baselines (XLM-R). Specifically, the large version of C-XLM achieves +2.9 p.p. in μ-F_1 and +3.2 p.p. in m-F_1 compared to the large version of XLM-R.

ContractNER: Similarly, our C-XLM models outperform the corresponding large and base baselines by approx. 0.5 p.p. in μ-F_1. In addition, m-F_1 is higher in our large model by 0.9 p.p., while base models have identical results. Again, the small version of C-XLM is competitive to the baseline.

In general trends, we observe that larger models outperform smaller ones in most cases, and domain-specific models outperform generic ones, while using a sunstantially smaller (4×) vocabulary and be significantly less (63×) pre-trained. The largest relative differences occur in MULTIEURLEX, a

20-class multi-label classificationtask, and CNLI, a sentence pair classification task.

Language Parity: Figure 3 provides information through radar plots, about scores per language for each variant of C-XLM. We generally observe that performance varies across languages (e.g., models perform better in English compared to German), while also language performance disparity varies across models (depicted as differently shaped webs), and across datasets (e.g., models are better in English compared to Italian in MultiEURLEX, but the opposite is true for UNFAIR-ToS).[12]

We cross out representation disparity as a possible explanation, since training data equally represent all languages (equal number of training examples). Interestingly, pre-training (MLM) accuracy also does not correlate with the down-stream performance. Based on the aforementioned points, we can only hypothesize that other qualitative characteristics (idiosyncrasies of a language in a specific context/domain) are responsible for perfomance disparities in-between languages.

Algorithm 1 Gradual Compression

if Teacher Size >> Student Size **then**
    S0: Distill *model* to *teacher assistant*
S1: Prune *model vocabulary* and
fine-tune for 1-3 epochs (if needed).
S2: Prune *model depth* and distill.
S3: Prune *model width* and re-distill.
S4.1: Optimize computational graph.
S4.2: Apply 8-bit dynamic quantization.

---

5 Model Compression

5.1 Methodology

To compress and accelerate the inference of fine-tuned transformer-based models we adopt *gradual compression*, a pipeline that combines structured pruning, knowledge distillation, and post-training quantization to progressively reach the desired compression rate, summarized in Algorithm 1.[13]

Step 0 — Teacher Assistant: In case the teacher is very large and the desired compression rate is high (e.g., reducing the large version of C-XLM to the tiny one), teacher assistants (Mirzadeh et al., 2020) are used to make the transition smoother.

---

[12]Refer to Appendix B for detailed results.
[13]See additional details and results from preliminary experiments in Appendix B.

Step 1 — Vocabulary Pruning: The first step is to reduce the model's vocabulary. Tokens that do not appear in the training dataset of the down-stream task are removed. Furthermore, using information from the tokenizer's merges, the merge of two tokens that exist in the training dataset and individual tokens that form a merge that, also, exists in the training dataset are kept as well. After the redundant tokens are removed, the embedding matrix is reshaped and the new model, if necessary, is fine-tuned for 1-3 epochs, to restore its original performance. The intuition behind vocabulary reduction is that some word embeddings that were learned during pre-training might not be useful for a specific down-stream task, since such words are rare and their word embeddings would not get updated during fine-tuning, if they did not exist in the training set (e.g., some words of a multilingual model would be redundant for a monolingual task).

Step 2 — Depth Pruning: The second step is to reduce the model's *depth* via knowledge distillation. Similarly to Sun et al. (2019), we find that using the weights of the first *k* layers from the teacher's original pre-trained (not fine-tuned) language model produces the most consistent results. In our implementation, the KL-divergence between the (softened) teacher's and student's predicted probabilities is chosen as the distillation loss function. Across all tasks, the distillation loss is, also, linearly combined with the original loss. For the multi-label classification task, the cross-entropy loss is replaced by a binary cross-entropy loss, again with the (softened) teacher's and student's probabilities as inputs, whereas for the regression task it is replaced by the mean squared error between the teacher's and student's output logits (Ba and Caruana, 2014).

Step 3 — Width Pruning: Once the fine-tuned teacher's knowledge is distilled to the student model, structured pruning is applied to reduce the student's *width*. In particular, using *TextPruner* (Yang et al., 2022), the top *n* neurons from the intermediate fully-connected layers and the top *a* attention heads from the multi-head attention layers that have the smallest impact on the expected loss are iteratively removed (Michel et al., 2019; Prasanna et al., 2020). The pruned student model is re-distilled to restore its original performance. Although unstructured pruning (Han et al., 2015; Sanh et al., 2020; Louizos et al., 2018) would probably lead to higher compression rates with smaller

performance loss, we choose structured pruning to ensure that the compressed model's inference speed is also accelerated.

Step 4 — Graph Optimization & Model Quantization: For the final step, the student's weights are quantized to 8-bits, using post-training dynamic quantization. However, although 8-bit quantization will reduce the memory footprint by approximately 4x, without specialized hardware there will be hardly any inference time speed-up. Thus, before quantizing the student model, using *ONNX* (Bai et al., 2019), its computational graph is optimized, which can provide hardware-independent acceleration (Li et al., 2021). In particular, constant folding –where constant expressions are statically pre-computed–, redundant node elimination –where redundant nodes such as identities are removed without changing the graph structure– and operation fusion –where multiple smaller nodes are fused into one, reducing in this way launch and synchronization overhead (Vasilache et al., 2018)– are applied.

Why gradual compression? Although gradual compression can be more time-consuming than, for example, distilling the teacher's knowledge in a student with a smaller predefined size, it offers more flexibility and control over the whole compression process. When the desired compression rate is reached gradually, one could better balance the performance/compression-rate trade-off.

If for example the model is sensitive to reducing the depth, one could prune the width more aggressively and vice versa. Since the model will only be compressed once before deployed, it is important to ensure that the productionized model will perform as well as possible, thus, devoting more time to take careful steps should not be a concern.

5.2 Compression Results

For each down-stream task, the goal is to produce compressed versions of the large and base C-XLM that can outperform the fine-tuned small and tiny variants of C-XLM, while being smaller and faster in terms of memory and inference speed. Using Gradual Compression (GC), the final compressed versions with the small version of C-XLM as a reference comprise 6 Transformer blocks, 24 attention heads, and 1024 units, whereas the compressed versions with the tiny one as a reference consist of 3 blocks, 12 heads and 512 units.

Model	MultiEURLEX		UNFAIR-ToS		CNLI		Obligations		ContractNER	
	μ-F$_1$	m-F$_1$	Acc.	MAE	μ-F$_1$	m-F$_1$	μ-F$_1$	m-F$_1$	μ-F$_1$	m-F$_1$
Top Bound - Performance "Ceiling"										
C-XLM (large)	78.4	65.4	89.7	0.14	85.3	83.0	91.8	90.6	93.2	94.6
Gradual Compression — Reference C-XLM (small)										
C-XLM (small) (FT)	72.3	54.7	**85.4**	**0.20**	79.7	77.0	90.4	89.0	90.1	92.4
C-XLM (small) (KD)	73.3	54.7	81.1	0.25	80.2	78.1	90.1	89.1	91.0	93.1
C-XLM (large) (GC)	**74.2**	**60.4**	83.7	0.21	**84.5**	**83.1**	**92.2**	**91.3**	**92.2**	**93.3**
Gradual Compression — Reference C-XLM (tiny)										
C-XLM (tiny) (FT)	66.5	46.1	78.2	0.27	70.2	69.2	88.7	87.4	87.2	89.3
C-XLM (tiny) (KD)	64.0	42.0	76.7	0.30	75.3	74.3	89.1	88.1	**87.7**	90.1
C-XLM (large) (GC)	**73.2**	**57.0**	**79.6**	**0.25**	**80.7**	**79.2**	**91.9**	**90.7**	87.6	**90.2**

Table 3: Model compression results across down-stream tasks. We report the performance for two baselines: (a) fine-tuning the reference pre-trained C-XLM model (FT), and (b) Knowledge Distillation and Vocabulary Pruning. where the student is the reference pre-trained C-XLM (KD); alongside the performance of fully gradually compressed (GC) models, i.e., pruned, distilled and quantized (P+KD+Q). We report the model's performance across the incremental compression steps (S) presented in Section 5.1 in the Appendix (Table 9).

The large C-LXM used as the teacher is substantially larger (20-40×) compared to the reference models. To ensure that the transition to compressed versions is smooth, we first distill it using as student the base version of C-XLM, to create a *teacher assistant* (Mirzadeh et al., 2020). In every incremental step of GC where knowledge distillation is applied, the learning rate, temperature and *a* (the original and distillation loss weighing) are tuned using grid search. Our GC compression pipeline is also compared with a variant of *Pre-trained Distillation* (Turc et al., 2019), where the teacher's (or its assistant's) knowledge is distilled directly to the reference (smaller) pre-trained model.

Results are presented in Table 3. We observe that the compressed versions of the large C-XLM model (GC) produced by the full-scale compression pipeline introduced in Section 5.1 always outperform both the respective fine-tuned (FT) models of the smaller versions of C-XLM and the distilled (KD) ones with a single exception in UNFAIR-ToS. Similar results can be derived when we consider the base version of C-XLM as the teacher model. Results are presented in Appendix B.

The largest relative differences are observed in the setting where we use the tiny model as a reference, which indicates that gradual compression is very effective when higher compression rates are being considered. Interestingly, in the Obligation extraction task, the compressed models are able to outperform the teacher (Upper Bound).

Model	Performance Loss	Compression Rate	Inference Acceleration
Reference C-XLM (small)			
FT	-4.1 p.p.	17.4×	34.1×
KD	-4.5 p.p.	21.0×	36.2×
GC	**-2.3 p.p.**	**41.8×**	**65.5×**
Reference C-XLM (tiny)			
FT	-9.5 p.p.	40.3×	87.9×
KD	-9.1 p.p.	**50.9×**	94.2×
GC	**-5.1 p.p.**	**50.9×**	**169.8×**

Table 4: Averaged performance and efficiency statistics for each model across all tasks.

Results can be vastly improved if a more fine-grained network-architecture search is conducted. For example, in some of the tasks, the largest performance drop occurs during the second step (depth pruning). This could be prevented if few additional layers remain, in favor of aggressive width pruning (step 3).[14] However, the goal of our experiments was to produce competitive results among all tasks, even with the constraint of using shared predefined network specifications.

5.3 Efficiency Considerations

In Table 4, we present aggregated (averaged) statistics in terms of efficiency.[14] With the small version of C-XLM as a reference, GC produces models that are 41.8× smaller and 65.5× faster, while losing

---

[14]The size and inference speed of models produced at each compression step are reported in Appendix B.4.
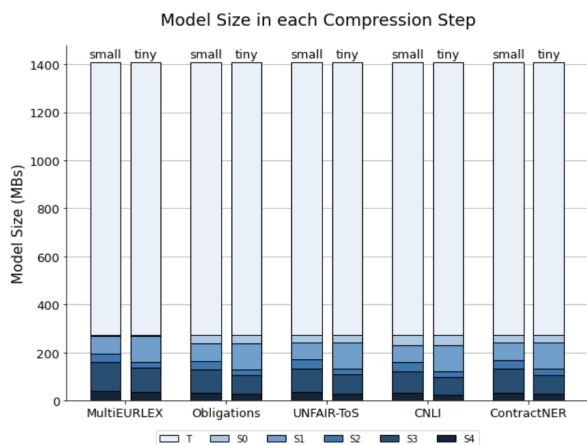
Figure 4: Model size (MB) in each compression step (S) in relevance to the original model, C-XLM (large).

only 2.3 p.p. of performance on average. On the other hand, the fine-tuned (FT) or distilled (KD) models have a larger performance drop (by approx. 1-2 p.p.) compared to the GC versions which are also substantially (almost 2×) faster on average.

With the tiny version of C-XLM as a reference, GC can produce models that are, on average, 50.9x smaller and 169.8x faster, while losing 5.1 p.p. of performance on average. The fine-tuned (FT) or distilled (KD) models have now substantially larger performance drop (9 p.p.) highlighting the benefits of GC in an extreme-compression setting.

In Figure 4, we present the model size reduction across the incremental GC steps (S0-S4). The largest size reduction in both settings (small, tiny) is observed in the quantization step (S4), if we exclude the preliminary distillation step to create the teacher's assistant (S0), which reduces the original model size approx. 4×.[15]

6 Related Work

6.1 Transformer-based LMs

Devlin et al. (2019) are the first to pre-train transformer-based language models (BERT) on large corpora that achieving state-of-the-art results in generic NLP benchmarks ((Wang et al., 2019b,a). One year later, Liu et al. (2019) argued that BERT was significantly under-trained and introduced RoBERTa (Robustly optimized BERT) using improved pre-training settings (more data, larger mini-batches, dynamic masking, and a larger vocabulary) leading to new state-of-the-art results.

Moreover, multilingually pre-trained models (Conneau and Lample, 2019; Conneau et al., 2020) have been developed using a shared vocabulary, which can later fine-tuned across several languages. These models have also shown to have exceptional zero-shot cross-lingual capabilities, a direction that we do not investigate in this work.

In the NLP literature, domain-specific models outperform generic ones in domain-specific benchmarking. Lee et al. (2019) created BioBERT by further pre-training BERT of (Devlin et al., 2019) on biomedical corpora. n the same manner, Alsentzer et al. (2019) further pre-trained BioBERT on clinical notes, releasing ClinicalBERT. Similarly, Beltagy et al. (2019) pretrained BERT model on scientific publications called SciBERT, while Loukas et al. (2022) released SEC-BERT pre-trained on US financial filings.

In the legal domain, Chalkidis et al. (2020) released LegalBERT, a legal-oriented BERT model pre-trained on diverse English legal corpora, which outperform generic ones in most legal NLU benchmark (Chalkidis et al., 2022) as is CaseLawBERT of Zheng et al. (2021), a BERT model pre-trained solely on US case law. Recently, (Henderson* et al., 2022) released a new legal-oriented larger BERT model, which is also heavily biased towards legal proceedings in US-based jurisdictions.

6.2 Model Compression

Unstructured pruning was popularized by Han et al. (2015), who iteratively located and pruned connections whose weights were less than a pre-specified threshold and retrained the sparsed network. In later work, the idea of learning how to sparsify models during training was also proposed (Zhu and Gupta, 2017; Louizos et al., 2018; Sanh et al., 2020). For Transformer-base models, Sanh et al. (2020) argued that changes in weights during fine-tuning must be taken into consideration and proposed *movement pruning*.

On the other hand, structured pruning produces smaller (not sparse) models by removing attention heads (Voita et al., 2019; McCarley et al., 2019; Michel et al., 2019)), individual (McCarley et al., 2019; Prasanna et al., 2020) or blocks (Lagunas et al., 2021) of neurons from fully-connected layers in a structured manner. We follow this line of work, since structured pruning improves model compression in practice (deployment of smaller models), contrary to unstructured pruning which sparsify

---

[15]The size compression effect of steps S1-S4 is better depicted in Figure 5 in Appendix B where the base version of C-XLM is the teacher and thus S0 is omitted.

networks (deployment of sparse, but equally-sized models comparing to the original ones).

Another approach to reduce the memory footprint of neural networks is quantization, i.e., mapping the real-valued parameters and activations over a fixed set of discrete numbers to minimize the number of bits required to store them. When transformer-based models are quantized to 8-bits, the models' memory overhead is reduced approximately by 4x (Bondarenko et al., 2021), while the matrix multiplication computational cost can be reduced by 3.7x with the use of specialized hardware. Junczys-Dowmunt et al. (2018) and Bhandare et al. (2019)) applied 8-bit post training quantization to transformer-based models. Zafrir et al. (2019) and Fan et al. (2020) used quantization aware training to quantize transformer-based language models.

The last technique that is frequently used to compress transformer-based models is knowledge distillation. With knowledge distillation, a smaller (*student*) network is trained to mimic the behavior of a larger (*teacher*) network. In particular, instead of training the student network with the true labels, the teacher's predictions are used as a target (*response-based knowledge distillation*), which are usually "softened" to better capture similarities across classes (Hinton et al., 2015).

Along with the teacher's predictions, information from the teacher's intermediate states (*feature-based knowledge distillation*) such as hidden states (Sun et al., 2019), embeddings (Jiao et al., 2020) and attention distributions (Sun et al., 2020) have been used, an interesting direction that we do not explore in this work.

7 Conclusions

Following model development across all three incremental steps of the examined *pipelined* approach, we make the following observations:

(a) Larger models outperform smaller ones; the performance increase varies across tasks.

(b) Domain-specific models outperform generic ones, although gains are decreased considering much large models.

(c) Fully compressed (pruned, distilled, and quantized) models severely outperform equally sized distilled or fine-tuned models.

To conclude, our guidelines to LegalTech practitioners who aim to build effective, but also efficient models, can be summarized in four general points:

1. Pre-train large-scale domain-specific language models, if possible; in case there are no such models already available.

2. Fine-tune the largest possible model available based on your compute capabilities.

3. Compress the fine-tuned models to derive much smaller models that can efficiently be deployed in production; consider a suitable compression rate to balance the performance / efficiency trade-offs.

4. Follow a full-scale compression pipeline (Vocabulary Pruning, Parameter Pruning, Knowledge Distillation, Graph Optimization and Quantization) for best results.

Broader Impact and Ethics Considerations

In this sections, we would like to discuss the broader impact and ethical considerations with respect to the use of data, privacy issues and environmental considerations.

Use of Data In this work, we considered two sources of open publicly available data. The first source is legislation from EU (Chalkidis et al., 2021b) published by the EU Publication Office,[16] UK (Chalkidis and Søgaard, 2022) published the UK National Archives,[17] and US (Henderson* et al., 2022) published by the U.S. Government Publishing Office.[18] The second source is US contracts (Tuggener et al., 2020; Borchmann et al., 2020) published as exhibits in public filings at SEC-EDGAR.[19] As discussed in Henderson* et al. (2022), the content from these legal sources implicitly encodes privacy and toxicity rules since its content is handled by governments and courts, contrary to generic web material scraped from the web (Dodge et al., 2021).

In another note, many of these sources that we used to pre-train our C-XLM models, overlap with the benchmark datasets we used to evaluate the very same models, e.g., the MultiEURLEX dataset used both for pre-training and evaluation (Sections 3.1 and 4.1). As Krishna et al. (2022) recently showed using downstream datasets make surprisingly good up-stream (pre-training) corpora, if domain specificity and such applications is the goal, in contrast

---

[16]https://eur-lex.europa.eu/
[17]https://www.legislation.gov.uk/
[18]http://www.gpo.gov/
[19]https://www.sec.gov/edgar/

to heavy generalization across domains and acquirement of common knowledge.[20]

Environmental Considerations Modern large deep learning models are cost intensive financially to train, due to the cost of hardware, electricity - especially in these challenging times-, and cloud compute (Strubell et al., 2019). They are also environmentally expensive due to the operational carbon footprint, i.e., carbon emissions, (Dodge et al., 2022). It has been also demonstrated that the impact of deployment and inference can be equally or more harmful compared to training with regards to carbon emissions (Wu et al., 2022), hence effective counter-measures should be considered to compensate for the financial and environmental cost.

By compressing and accelerating larger models, the carbon footprint of inference can be significantly reduced as we show in Section 5.3; compensating in this way (on the long run) the environmental implications of large-scale training. Furthermore, by decreasing their memory requirements (model size, and architecture complexity), predictive models can be hosted on more environmentally-friendly infrastructure, e.g., moderate-compute cloud servers with low memory and processing power leading to a decreased energy footprint, contrary to high-end energy-intensive GPU-accelerated machines.

Privacy Considerations Privacy concerns are also a critical topic, especially in the legal-tech industry, since prospect users (law firms, companies, and civilians etc.) want to process large quantities of documents, many of which include confidential information (e.g., private contracts). While there are many directions to privacy preserving ML via differential privacy (Abadi et al., 2016; Klymenko et al., 2022) or federated learning (Ryffel et al., 2018), the problem of data leakage is practically unsolved, since the risks of sharing private documents are not considered and the responsibilities are transferred to data and cloud security.

Since highly accurate compressed models are able to be developed (Section 5.1), deployed and run on moderate-compute servers (Section 5.3), such technologies can be deployed on premises as an in-house solution on private clouds; or even run on the client side on server-client web infrastructures, eliminating the need for hosting data remotely or using API calls to remote cloud servers over the web, thus effectively contribute in a safer, more secure (private) AI.

Limitations

Based on our experiments, similarly to the literature, there no is free lunch with respect to model compression, and further compressing models takes a toll on performance. Experimenting with much larger models and examining their performance and potential for compression, following the line of work of Rae et al. (2021); Hoffmann et al. (2022) would be fascinating but we lack resources to built billion-parameter-sized models, while increasing resources would have a larger impact with respect to environmental considerations. Based on the findings of Hoffmann et al. (2022), our models are not under-trained, and exploring larger models would have to be followed by an analogous increase of pre-training data and compute.

Acknowledgments

References

Martin Abadi, Andy Chu, Ian Goodfellow, H. Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. 2016. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, CCS '16, page 308–318, New York, NY, USA. Association for Computing Machinery.

Emily Alsentzer, John Murphy, William Boag, Wei-Hung Weng, Di Jindi, Tristan Naumann, and Matthew McDermott. 2019. Publicly available clinical BERT embeddings. In *Proceedings of the 2nd Clinical Natural Language Processing Workshop*,

---

[20]Of course, we always consider fair evaluation practices, i.e., no access to the test subsets of evaluation datasets.

[21]https://innovationsfonden.dk/en
[22]https://sites.research.google/trc/about/

pages 72–78, Minneapolis, Minnesota, USA. Association for Computational Linguistics.

Jimmy Ba and Rich Caruana. 2014. Do deep nets really need to be deep? In *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc.

Junjie Bai, Fang Lu, Ke Zhang, et al. 2019. Onnx: Open neural network exchange. https://github.com/onnx/onnx.

Iz Beltagy, Kyle Lo, and Arman Cohan. 2019. SciBERT: A pretrained language model for scientific text. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3615–3620, Hong Kong, China. Association for Computational Linguistics.

Aishwarya Bhandare, Vamsi Sripathi, Deepthi Karkada, Vivek Menon, Sun Choi, Kushal Datta, and Vikram Saletore. 2019. Efficient 8-bit quantization of transformer neural machine language translation model. *arXiv preprint arXiv:1906.00532*.

Yelysei Bondarenko, Markus Nagel, and Tijmen Blankevoort. 2021. Understanding and overcoming the challenges of efficient transformer quantization. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7947–7969, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Łukasz Borchmann, Dawid Wisniewski, Andrzej Gretkowski, Izabela Kosmala, Dawid Jurkiewicz, Łukasz Szałkiewicz, Gabriela Pałka, Karol Kaczmarek, Agnieszka Kaliska, and Filip Graliński. 2020. Contract discovery: Dataset and a few-shot semantic retrieval challenge with competitive baselines. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4254–4268, Online. Association for Computational Linguistics.

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.

Ilias Chalkidis, Ion Androutsopoulos, and Achilleas Michos. 2017. Extracting contract elements. In *Proceedings of the 16th Edition of the International Conference on Articial Intelligence and Law*, ICAIL '17, page 19–28, New York, NY, USA. Association for Computing Machinery.

Ilias Chalkidis, Ion Androutsopoulos, and Achilleas Michos. 2018. Obligation and prohibition extraction using hierarchical RNNs. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 254–259, Melbourne, Australia. Association for Computational Linguistics.

Ilias Chalkidis, Manos Fergadiotis, and Ion Androutsopoulos. 2021a. MultiEURLEX - a multi-lingual and multi-label legal document classification dataset for zero-shot cross-lingual transfer. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Online.

Ilias Chalkidis, Manos Fergadiotis, and Ion Androutsopoulos. 2021b. MultiEURLEX - a multi-lingual and multi-label legal document classification dataset for zero-shot cross-lingual transfer. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6974–6996, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Ilias Chalkidis, Manos Fergadiotis, Prodromos Malakasiotis, Nikolaos Aletras, and Ion Androutsopoulos. 2020. LEGAL-BERT: The muppets straight out of law school. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2898–2904, Online.

Ilias Chalkidis, Abhik Jana, Dirk Hartung, Michael Bommarito, Ion Androutsopoulos, Daniel Katz, and Nikolaos Aletras. 2022. LexGLUE: A benchmark dataset for legal language understanding in English. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4310–4330, Dublin, Ireland. Association for Computational Linguistics.

Ilias Chalkidis and Anders Søgaard. 2022. Improved multi-label classification under temporal concept drift: Rethinking group-robust algorithms in a label-wise setting. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 2441–2454, Dublin, Ireland. Association for Computational Linguistics.

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*, pages 8440–8451, Online. Association for Computational Linguistics.

Alexis Conneau and Guillaume Lample. 2019. Cross-lingual language model pretraining. In *Advances in Neural Information Processing Systems*, volume 32. 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.

Jesse Dodge, Taylor Prewitt, Remi Tachet des Combes, Erika Odmark, Roy Schwartz, Emma Strubell, Alexandra Sasha Luccioni, Noah A. Smith, Nicole DeCario, and Will Buchanan. 2022. Measuring the carbon intensity of ai in cloud instances. In *2022 ACM Conference on Fairness, Accountability, and Transparency*, FAccT '22, page 1877–1894, New York, NY, USA. Association for Computing Machinery.

Jesse Dodge, Maarten Sap, Ana Marasović, William Agnew, Gabriel Ilharco, Dirk Groeneveld, Margaret Mitchell, and Matt Gardner. 2021. Documenting large webtext corpora: A case study on the colossal clean crawled corpus. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1286–1305, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Kasper Drawzeski, Andrea Galassi, Agnieszka Jablonowska, Francesca Lagioia, Marco Lippi, Hans Wolfgang Micklitz, Giovanni Sartor, Giacomo Tagiuri, and Paolo Torroni. 2021. A corpus for multilingual analysis of online terms of service. In *Proceedings of the Natural Legal Language Processing Workshop 2021*, pages 1–8, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Angela Fan, Pierre Stock, Benjamin Graham, Edouard Grave, Rémi Gribonval, Herve Jegou, and Armand Joulin. 2020. Training with quantization noise for extreme model compression. *arXiv preprint arXiv:2004.07320*.

Ivan Habernal, Daniel Faber, Nicola Recchia, Sebastian Bretthauer, Iryna Gurevych, Indra Spiecker genannt Döhmann, and Christoph Burchard. 2022. Mining Legal Arguments in Court Decisions. *arXiv preprint*.

Song Han, Jeff Pool, John Tran, and William Dally. 2015. Learning both weights and connections for efficient neural network. In *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc.

Peter Henderson*, Mark S. Krass*, Lucia Zheng, Neel Guha, Christopher D. Manning, Dan Jurafsky, and Daniel E. Ho. 2022. Pile of law: Learning responsible data filtering from the law and a 256gb open-source legal dataset.

Geoffrey Hinton, Oriol Vinyals, and Jeffrey Dean. 2015. Distilling the knowledge in a neural network. In *NIPS Deep Learning and Representation Learning Workshop*.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katie Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, Jack W. Rae, Oriol Vinyals, and Laurent Sifre. 2022. Training compute-optimal large language models.

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.

Marcin Junczys-Dowmunt, Kenneth Heafield, Hieu Hoang, Roman Grundkiewicz, and Anthony Aue. 2018. Marian: Cost-effective high-quality neural machine translation in c++. In *Proceedings of the 2nd Workshop on Neural Machine Translation and Generation*, pages 129–135.

Oleksandra Klymenko, Stephen Meisenbacher, and Florian Matthes. 2022. Differential privacy in natural language processing the story so far. In *Proceedings of the Fourth Workshop on Privacy in Natural Language Processing*, pages 1–11, Seattle, United States. Association for Computational Linguistics.

Yuta Koreeda and Christopher Manning. 2021. ContractNLI: A dataset for document-level natural language inference for contracts. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 1907–1919, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Kundan Krishna, Saurabh Garg, Jeffrey P. Bigham, and Zachary C. Lipton. 2022. Downstream datasets make surprisingly good pretraining corpora.

François Lagunas, Ella Charlaix, Victor Sanh, and Alexander Rush. 2021. Block pruning for faster transformers. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10619–10629, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

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

Mingzhen Li, Yi Liu, Xiaoyan Liu, Qingxiao Sun, Xin You, Hailong Yang, Zhongzhi Luan, Lin Gan, Guangwen Yang, and Depei Qian. 2021. The deep learning compiler: A comprehensive survey. *IEEE Transactions on Parallel and Distributed Systems*, 32(3):708–727.

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

Christos Louizos, Max Welling, and Diederik P. Kingma. 2018. Learning sparse neural networks through l0 regularization. In *International Conference on Learning Representations*.

Lefteris Loukas, Manos Fergadiotis, Ilias Chalkidis, Eirini Spyropoulou, Prodromos Malakasiotis, Ion Androutsopoulos, and Georgios Paliouras. 2022. FiNER: Financial numeric entity recognition for XBRL tagging. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4419–4431, Dublin, Ireland. Association for Computational Linguistics.

JS McCarley, Rishav Chakravarti, and Avirup Sil. 2019. Structured pruning of a bert-based question answering model. *arXiv preprint arXiv:1910.06360*.

Paul Michel, Omer Levy, and Graham Neubig. 2019. Are sixteen heads really better than one? In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Seyed Iman Mirzadeh, Mehrdad Farajtabar, Ang Li, Nir Levine, Akihiro Matsukawa, and Hassan Ghasemzadeh. 2020. Improved knowledge distillation via teacher assistant. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(04):5191–5198.

Sai Prasanna, Anna Rogers, and Anna Rumshisky. 2020. When BERT Plays the Lottery, All Tickets Are Winning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3208–3229, Online. Association for Computational Linguistics.

Alec Radford and Karthik Narasimhan. 2018. Improving language understanding by generative pre-training.

Jack W. Rae, Sebastian Borgeaud, Trevor Cai, Katie Millican, Jordan Hoffmann, H. Francis Song, John Aslanides, Sarah Henderson, Roman Ring, Susannah Young, Eliza Rutherford, Tom Hennigan, Jacob Menick, Albin Cassirer, Richard Powell, George van den Driessche, Lisa Anne Hendricks, Maribeth Rauh, Po-Sen Huang, Amelia Glaese, Johannes Welbl, Sumanth Dathathri, Saffron Huang, Jonathan Uesato, John Mellor, Irina Higgins, Antonia Creswell, Nat McAleese, Amy Wu, Erich Elsen, Siddhant M. Jayakumar, Elena Buchatskaya, David Budden, Esme Sutherland, Karen Simonyan, Michela Paganini, Laurent Sifre, Lena Martens, Xiang Lorraine Li, Adhiguna Kuncoro, Aida Nematzadeh, Elena Gribovskaya, Domenic Donato, Angeliki Lazaridou, Arthur Mensch, Jean-Baptiste Lespiau, Maria Tsimpoukelli, Nikolai Grigorev, Doug Fritz, Thibault Sottiaux, Mantas Pajarskas, Toby Pohlen, Zhitao Gong, Daniel Toyama, Cyprien de Masson d'Autume, Yujia Li, Tayfun Terzi, Vladimir Mikulik, Igor Babuschkin, Aidan Clark, Diego de Las Casas, Aurelia Guy, Chris Jones, James Bradbury, Matthew Johnson, Blake A. Hechtman, Laura Weidinger, Iason Gabriel, William S. Isaac, Edward Lockhart, Simon Osindero, Laura Rimell, Chris Dyer, Oriol Vinyals, Kareem Ayoub, Jeff Stanway, Lorrayne Bennett, Demis Hassabis, Koray Kavukcuoglu, and Geoffrey Irving. 2021. Scaling language models: Methods, analysis & insights from training gopher. *CoRR*, abs/2112.11446.

Théo Ryffel, Andrew Trask, Morten Dahl, Bobby Wagner, Jason Mancuso, Daniel Rueckert, and Jonathan Passerat-Palmbach. 2018. A generic framework for privacy preserving deep learning. *CoRR*, abs/1811.04017.

Victor Sanh, Thomas Wolf, and Alexander Rush. 2020. Movement pruning: Adaptive sparsity by fine-tuning. In *Advances in Neural Information Processing Systems*, volume 33, pages 20378–20389. Curran Associates, Inc.

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 (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

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

Siqi Sun, Yu Cheng, Zhe Gan, and Jingjing Liu. 2019. Patient knowledge distillation for BERT model compression. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4323–4332, Hong Kong, China. Association for Computational Linguistics.

Zhiqing Sun, Hongkun Yu, Xiaodan Song, Renjie Liu, Yiming Yang, and Denny Zhou. 2020. MobileBERT: a compact task-agnostic BERT for resource-limited devices. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2158–2170, Online. Association for Computational Linguistics.

Chul Sung, Tejas Dhamecha, Swarnadeep Saha, Tengfei Ma, Vinay Reddy, and Rishi Arora. 2019. Pre-training BERT on domain resources for short answer grading. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language*

Processing and the 9th International Conference on Natural Language Processing (EMNLP-IJCNLP), pages 6071–6075, Hong Kong, China. Association for Computational Linguistics.

Don Tuggener, Pius von Däniken, Thomas Peetz, and Mark Cieliebak. 2020. LEDGAR: A large-scale multi-label corpus for text classification of legal provisions in contracts. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 1235–1241, Marseille, France. European Language Resources Association.

Iulia Turc, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Well-read students learn better: The impact of student initialization on knowledge distillation. *CoRR*, abs/1908.08962.

Nicolas Vasilache, Oleksandr Zinenko, Theodoros Theodoridis, Priya Goyal, Zachary DeVito, William S. Moses, Sven Verdoolaege, Andrew Adams, and Albert Cohen. 2018. Tensor comprehensions: Framework-agnostic high-performance machine learning abstractions.

Elena Voita, David Talbot, Fedor Moiseev, Rico Sennrich, and Ivan Titov. 2019. Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5797–5808, Florence, Italy. Association for Computational Linguistics.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2019a. Superglue: A stickier benchmark for general-purpose language understanding systems. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019b. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *International Conference on Learning Representations*, New Orleans, Louisiana, USA.

Carole-Jean Wu, Ramya Raghavendra, Udit Gupta, Bilge Acun, Newsha Ardalani, Kiwan Maeng, Gloria Chang, Fiona Aga, Jinshi Huang, Charles Bai, Michael Gschwind, Anurag Gupta, Myle Ott, Anastasia Melnikov, Salvatore Candido, David Brooks, Geeta Chauhan, Benjamin Lee, Hsien-Hsin Lee, Bugra Akyildiz, Maximilian Balandat, Joe Spisak, Ravi Jain, Mike Rabbat, and Kim Hazelwood. 2022. Sustainable ai: Environmental implications, challenges and opportunities. In *Proceedings of Machine Learning and Systems*, volume 4, pages 795–813.

Stratos Xenouleas, Alexia Tsoukara, Giannis Panagiotakis, Ilias Chalkidis, and Ion Androutsopoulos. 2022. Realistic zero-shot cross-lingual transfer in legal topic classification. In *Proceedings of the 12th*

EETN Conference on Artificial Intelligence (SETN 2022).

Ziqing Yang, Yiming Cui, and Zhigang Chen. 2022. TextPruner: A model pruning toolkit for pre-trained language models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 35–43, Dublin, Ireland. Association for Computational Linguistics.

Ofir Zafrir, Guy Boudoukh, Peter Izsak, and Moshe Wasserblat. 2019. Q8bert: Quantized 8bit bert. In *2019 Fifth Workshop on Energy Efficient Machine Learning and Cognitive Computing-NeurIPS Edition (EMC2-NIPS)*, pages 36–39. IEEE.

Lucia Zheng, Neel Guha, Brandon R. Anderson, Peter Henderson, and Daniel E. Ho. 2021. When does pretraining help?: assessing self-supervised learning for law and the casehold dataset of 53,000+ legal holdings. *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Law*.

Michael Zhu and Suyog Gupta. 2017. To prune, or not to prune: exploring the efficacy of pruning for model compression. Technical report, Google.

A Experimentail Details

Devlin et al. (2019) suggested a hyperparameter tuning, that was adopted in many papers (Lee et al., 2019; Beltagy et al., 2019; Alsentzer et al., 2019; Sung et al., 2019). This hyper-parameter tuning included a light grid search in learning rate $\in$ {2e-5, 3e-5, 4e-5, 5e-5}, the number of training epochs $\in$ {3, 4}, and the batch size $\in$ {16, 32} with a fixed dropout rate of 0.1. In our research, we tune each variation of our model based on a grid-search of learning rate on the following range $\in$ {1e-4, 3e-4, 1e-5, 3e-5, 5e-5, 1e-6}. The batch size is fixed to 16, and the dropout rate at 0.1. The max sequence length is fixed to 512 for MultiEURLEX and ContractNLI, 256 for ContractNER and 128 for Contract-Obligations and UNFAIR-ToS based on the training subset statistics. Lastly, Chalkidis et al. (2020) found that some models may underfit for 4 epochs. Hence, following their work, we use early stopping based on validation loss up to 20 maximum train epochs with a patience of 3 epochs.

In every incremental step of GC where knowledge distillation is applied, learning rate, temperature and a (the original and distillation loss weighing) are tuned using grid search. in the hyper-parameter spaces of [1e-5, 3e-5, 5-e5, 7e-5, 1e-4], [1, 5, 10, 15] and [0.1, 0.3, 0.6], respectively.

Corpus	Tokens per Language										
	EN	EL	DE	FR	ES	IT	NL	PL	PT	RU	All
Contracts	125.3M	111.1M	103.3M	121.7M	122.5M	113.4M	110.7M	89.8M	111.3M	89.6M	1.1B
Regulations	178.7M	71.4M	62.8M	74.0M	77.1M	70.1M	71.2M	31.9M	71.5M	-	708.7M
All	304M	182.5M	166.1M	195.7M	199.6M	183.5M	181.9M	121.7M	182.8M	89.6M	1.8B

Table 5: Total tokens used per language per pre-training corpus.

Model	CNLI		Obligations		ContractNER	
	μ-F_1	m-F_1	μ-F_1	m-F_1	μ-F_1	m-F_1
Baselines						
C-XLM Base (Ceiling Baseline)	84.0	82.1	91.2	90.4	92.9	93.8
C-XLM Tiny (Bottom Baseline)	70.2	69.2	88.7	87.4	87.2	89.3
S1: Vocab Pruning						
S1.1: Prune vocabulary at random	79.4	77.2	88.7	87.6	88.5	83.2
S1.2: Prune vocabulary based on training data	**84.8**	**82.9**	**91.7**	**90.6**	**93.2**	**94.1**
Input Model: S1.2 –> S2: Pruning Depth (Layers) + KD						
S2.1: Prune layers at random	79.4	77.1	87.6	87.6	88.7	90.3
S2.2: Prune last N layers	80.6	78.8	**92.1**	**91.2**	**88.8**	**91.2**
S2.3: Prune first N layers	77.8	76.4	90.8	89.2	78.9	83.6
S2.4: Prune every second layer	**81.0**	**79.6**	89.9	88.5	88.6	89.4
S2.5: Prune layers with mimimum pair-wise distance	75.6	72.6	91.5	90.5	85.6	87.3
Input Model: S2.2 –> S3: Pruning Width (Heads + FF) + KD						
S3.1: Prune heads and FF at random	77.9	75.8	90.6	89.3	82.6	87.1
S3.2: Prune heads and FF based on the exp. sensitivity(L)	**78.0**	**76.1**	**91.6**	**90.7**	**89.5**	**92.3**

Table 6: Preliminiary Experiments to determine which settings produce the most consistent results.

B   Additional Results

B.1   Additional Pre-training Results

This section provides additional results regarding the pre-training process. Table 5 displays the number of tokens that were included in contracts and regulations for each language. It should be noted that during the pre-training, 100% of regulations and 20% of translated contracts (100% of English contracts used) were used. Finally, Table 15 presents the evaluation loss and accuracy scores of the pre-training of the masked language models. This table provides the overall scores, along with scores for each document type, language and document type/language.

B.2   Model Compression Preliminary Experiments

Before each model was compressed, some preliminary experiments were conducted to determine which setting in each compression step produces the most consistent results.

For the first step (Vocabulary Reduction), our
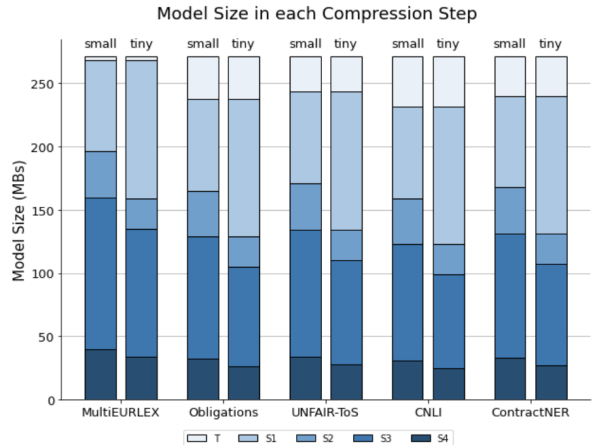
Figure 5: Model size (MB) in each compression step (S) in relevance to the original model, C-XLM (large).

proposed method, where the model's vocabulary is pruned based on information from the training data and the tokenizer's merges, was compared with a random baseline, i.e., the vocabulary is randomly pruned. In both settings, the exact same percentage of tokens were kept, and as expected, the random baseline led to performance deterioration.

Task	Reduction of Tokens	Removed Params (C-XLM Large)	Removed Params (C-XLM Base)
MultiEURLEX	2.52%	1,653,760	826,880
Obligations	26.98%	17,679,360	8,839,680
UNFAIR-ToS	22.54%	14,774,272	7,387,136
ContractNLI	31.83%	20,858,880	10,429,440
ContractNER	24.17%	15,839,232	7,919,616

Table 7: Percentage of usable tokens and parameter reduction after vocabulary pruning for each task.

Model	MultiEURLEX	UNFAIR-ToS	CNLI	Obligations	ContractNER
XLM-R (Base)	1e-5	3e-5	1e-5	1e-5	5e-5
XLM-R (Large)	3e-5	1e-5	1e-5	1e-6	5e-5
C-XLM (Tiny)	1e-4	3e-4	1e-4	1e-4	1e-4
C-XLM (Small)	1e-4	5e-5	1e-4	5e-5	5e-5
C-XLM (Base)	5e-5	3e-5	5e-5	3e-5	3e-5
C-XLM (Large)	5e-5	5e-5	3e-5	1e-5	5e-5

Table 8: Optimal Learning Rates per downstream task across all models.

For the second step (Depth Pruning), 9 out of the 12 transformer blocks of the base (language) model were removed and the pruned model was trained using knowledge distillation. Five different settings were tested. Encoder blocks were pruned by: (i) pruning blocks at random, (ii) pruning the last 9 blocks, (iii) pruning the first 9 blocks, (iv) pruning every second block, and (v) pruning blocks with the minimum pair-wise distance. For setting (v), the mean absolute error and the cosine similarity of the CLS tokens of each encoder block were used as the metrics (except for the ContractNER task, were the average of all tokens was used instead of the CLS one). We found that among all settings, copying the weights of the first encoder blocks of the original pre-trained language model produced the most consistent results.

For the third step (Width Pruning), the model of the second setting from the second step was pruned down to 12 attention heads (in total) and 512 neurons in the intermediate fully-connected layers. Two settings were examined: random pruning and pruning based on the expected loss when each attention head/neuron was iteratively removed (Michel et al., 2019; Prasanna et al., 2020). Just like in step 1, the random baseline performed worse. Results are summarized in Table 6.

B.3 Additional Fine-tuning Results

This section presents some additional results regarding the fine-tuning process. Table 8 displays the optimal learning rates for each model variation and baseline that were used during the fine-tuning process for every different task. Each one was selected through grid search, between learning rates $\in$ {1e-4, 3e-4, 1e-5, 3e-5, 5e-5, 1e-6}. We observe that smaller models favor larger learning rates, i.e., 1e-4 and 5e-5 in most cases, while larger models favor smaller learning rates, i.e., 1e-5 and 3e-5. Additionally, the upper parts of Tables 12 and 13 present the μ-F$_1$ and m-F$_1$ scores of the fine tuned models per language, respectively, for MultiEURLEX task. Lastly, the upper part of Table 14 presents the Mean Absolute Error (MAE) and accuracy scores of the fine-tuned models, for UNFAIR-ToS task per language.

B.4 Additional Compression Results

In this section, some additional experimental results are presented. First, the percentage of usable tokens and the parameter reduction of both large and base models from the vocabulary pruning step for each task can be found in Table 7. In Table 11, the model size (in MBs) and the average inference time (in seconds) of a 32-batch (on CPU) across all incremental compression steps and baselines are presented. Inference benchmarking was conducted using a modern mid-range RYZEN

7 4700ᴜ. In Table 10, model compression results can be found, across all down-stream task, when the Base C-XLM is used as a teacher. Lastly, Tables 12, 13 and 14 summarize the μ-F_1, m-F_1 of MultiEURLEX and results of UNFAIR-ToS tasks, across all languages for each incremental compression step and baselines.

Model	MultiEURLEX		UNFAIR-ToS		CNLI		Obligations		ContractNER	
	μ-F$_1$	m-F$_1$	Acc.	MAE	μ-F$_1$	m-F$_1$	μ-F$_1$	m-F$_1$	μ-F$_1$	m-F$_1$
Top Bound - Performance "Ceiling"										
C-XLM (large)	78.4	65.4	89.7	0.14	85.3	83.0	91.8	90.6	93.2	94.6
Step 0 (TA-KD)	75.2	63.0	88.6	0.16	84.6	82.2	92.8	91.8	93.7	94.9
Gradual Compression — Reference C-XLM (small)										
Step 1 (VP+KD)	75.1	62.9	88.5	0.18	84.9	82.9	92.8	91.8	93.6	94.9
Step 2 (DP+KD)	74.4	54.3	83.5	0.22	84.0	81.8	92.6	91.6	92.4	93.2
Step 3 (WP+KD)	73.8	60.8	83.5	0.21	84.5	83.0	92.3	91.4	92.1	93.2
Step 4 (GO+Q)	**74.2**	**60.4**	83.7	0.21	**84.5**	**83.1**	**92.2**	**91.3**	**92.2**	**93.3**
C-XLM (small) (FT)	72.3	54.7	**85.4**	**0.20**	79.7	77.0	90.4	89.0	90.1	92.4
C-XLM (small) (KD)	73.3	54.7	81.1	0.25	80.2	78.1	90.1	89.1	91.0	93.1
Gradual Compression — Reference C-XLM (tiny)										
Step 1 (VP+KD)	75.1	62.9	88.5	0.18	84.9	82.9	92.8	91.8	93.6	94.9
Step 2 (DP+KD)	72.6	58.2	80.6	0.24	80.9	79.3	91.1	90.0	89.7	92.1
Step 3 (WP+KD)	72.5	58.0	79.9	0.25	80.9	79.4	91.6	90.5	87.6	90.4
Step 4 (GO+Q)	**73.2**	**57.0**	**79.6**	**0.25**	80.7	79.2	**91.9**	**90.7**	87.6	**90.2**
C-XLM (tiny) (FT)	66.5	46.1	78.2	0.27	70.2	69.2	88.7	87.4	87.2	89.3
C-XLM (tiny) (VP+KD)	64.0	42.0	76.7	0.30	75.3	74.3	89.1	88.1	**87.7**	90.1

Table 9: Model compression results across down-stream tasks. We report the model's performance across the incremental compression steps (S) presented in Section 5.1. We also report the performance for two baselines: (a) fine-tuning the reference pre-trained C-XLM model (FT), and (b) Knowledge Distillation and Vocabulary Pruning. where the student is the reference pre-trained C-XLM (KD).

Model	MultiEURLEX		UNFAIR-ToS		CNLI		Obligations		ContractNER	
	μ-F$_1$	m-F$_1$	Acc.	MAE	μ-F$_1$	m-F$_1$	μ-F$_1$	m-F$_1$	μ-F$_1$	m-F$_1$
Top Bound - Performance "Ceiling"										
C-XLM (Base)	75.3	59.4	87.3	0.18	84.0	82.1	91.2	90.4	92.9	93.8
Gradual Compression — Reference C-XLM (small)										
Step 1 (VP+KD)	74.9	60.3	88.2	0.17	84.8	82.9	91.7	90.6	93.2	94.1
Step 2 (DP+KD)	74.4	59.3	82.7	0.23	84.2	82.4	91.3	90.2	91.6	91.7
Step 3 (WP+KD)	73.8	61.5	83.2	0.23	84.8	83.7	92.8	91.5	92.7	94.1
Step 4 (GO+Q)	**73.7**	**61.7**	83.2	0.23	**84.5**	**83.1**	**92.7**	**91.4**	**92.7**	**93.7**
C-XLM (small) (FT)	69.9	51.7	**85.4**	**0.20**	79.7	77.0	90.4	89.0	90.1	92.4
C-XLM (small) (KD)	72.3	54.7	82.7	0.23	80.0	78.4	90.3	89	92.0	92.4
Gradual Compression — Reference C-XLM (tiny)										
Step 1 (VP+KD)	74.9	60.3	88.2	0.17	84.8	82.9	91.7	90.6	93.2	94.1
Step 2 (DP+KD)	71.2	55.7	81.8	0.22	79.1	77.0	92.1	91.2	89.0	90.6
Step 3 (WP+KD)	70.6	55.8	80.8	0.24	78.0	76.1	91.6	90.7	89.8	92.5
Step 4 (GO+Q)	**70.4**	**54.4**	**80.8**	**0.24**	**78.5**	**76.8**	**91.6**	**90.7**	**89.5**	**92.3**
C-XLM (tiny) (FT)	66.5	46.1	78.2	0.27	70.2	69.2	88.7	87.4	87.2	89.3
C-XLM (tiny) (KD)	66.3	45.3	75.3	0.31	74.6	72.2	86.8	85.6	87.7	89.9

Table 10: Model compression results across down-stream tasks when Base C-XLM is used as a teacher. We report the model's performance across the incremental compression steps (S) presented in Section 5.1. We also report the performance for two baselines: (a) fine-tuning the reference pre-trained C-XLM model (FT), and (b) Knowledge Distillation and Vocabulary Pruning. where the student is the reference pre-trained C-XLM (KD).

Model	MultiEURLEX		UNFAIR-ToS		CNLI		Obligations		ContractNER	
	Size	Time	Size	Time	Size	Time	Size	Time	Size	Time
Top Bound - Performance "Ceiling"										
C-XLM (large)	1,409	99.3	1,409	21.5	1,409	116.0	1,409	21.3	1409	45.6
Gradual Compression — Reference C-XLM (tiny)										
Step 1 (VP+KD)	268	16.1	243	3.6	232	20.2	238	3.3	240	7.4
Step 2 (DP+KD)	159	4.1	134	0.8	123	4.1	129	0.8	131	1.8
Step 3 (WP+KD)	135	1.8	110	0.4	99	1.8	105	0.3	107	0.8
Step 4 (GO+Q)	**34**	**0.8**	**28**	**0.1**	**25**	**0.8**	**26**	**0.1**	**27**	**0.3**
C-XLM (tiny) (FT)	35	1.3	35	0.2	35	2.0	35	0.2	35	0.5
C-XLM (tiny) (KD)	**34**	1.3	**28**	0.2	**25**	1.3	**26**	0.2	**27**	0.5
Gradual Compression — Reference C-XLM (small)										
Step 1 (VP+KD)	268	16.1	243	3.6	232	20.2	238	3.3	240	7.4
Step 2 (DP+KD)	196	8.3	171	1.8	159	8.3	165	1.6	168	3.5
Step 3 (WP+KD)	160	4.4	110	0.9	123	4.4	129	0.9	131	1.9
Step 4 (GO+Q)	**40**	**1.8**	**34**	**0.3**	**31**	**1.8**	**32**	**0.3**	**33**	**0.7**
C-XLM (small) (FT)	81	3.2	81	0.6	81	4.1	81	0.5	81	1.4
C-XLM (small) (KD)	80	3.2	67	0.6	61	3.2	65	0.5	66	1.3

Table 11: Model compression results across down-stream tasks. We report the model's size in MBs and average inference time, in seconds, of a 32-batch across the incremental compression steps (S) presented in Section 5.1. We also report the performance for two baselines: (a) fine-tuning the reference pre-trained C-XLM model (FT), and (b) Knowledge Distillation and Vocabulary Pruning. where the student is the reference pre-trained C-XLM (KD).

Model	MultiEURLEX per Language μ-F_1									
	EN	FR	DE	NL	IT	ES	PT	PL	EL	μ-F_1
Fine-tuned Models										
XLM-R (large)	80.6	78.8	76.1	76.1	78.5	80.1	75.7	75.8	78.7	77.8
XLM-R (base)	77.6	76.9	73.7	74.3	76.3	75.0	75.7	73.2	75.6	75.3
C-XLM (large)	80.5	77.7	76.4	76.8	77.4	79.7	78.3	76.9	82.0	78.4
C-XLM (base)	77.5	76.4	72.3	74.1	75.6	77.4	75.0	72.9	76.5	75.3
C-XLM (small)	69.0	65.7	65.1	62.2	66.4	66.4	66.6	65.3	71.6	72.3
C-XLM (tiny)	69.0	65.7	65.1	62.2	66.4	66.4	66.6	65.3	71.6	66.5
Gradual Compression — Reference C-XLM (tiny)										
Step 1 (VP+KD)	78.0	75.9	74.0	73.1	75.2	78.5	75.4	73.4	72.0	75.1
Step 2 (DP+KD)	75.3	72.3	67.2	71.4	73.9	75.9	73.3	72.3	71.9	72.6
Step 3 (WP+KD)	74.1	70.7	69.2	71.9	74.4	74.2	72.9	71.8	73.1	72.5
Step 4 (GO+Q)	74.2	70.8	70.9	72.8	74.7	74.2	73.7	72.5	73.5	73.2
C-XLM (tiny) (KD)	64.9	63.3	60.8	63.2	66.3	63.1	63.8	65.0	65.7	64.0
Gradual Compression — Reference C-XLM (small)										
Step 1 (VP+KD)	78.0	75.9	74.0	73.1	75.2	78.5	75.4	73.4	2.0	75.1
Step 2 (DP+KD)	76.5	75.6	71.0	71.4	75.9	77.4	75.5	73.0	73.1	74.4
Step 3 (WP+KD)	75.8	75.2	70.4	70.5	74.4	75.6	73.9	74.2	74.7	73.8
Step 4 (GO+Q)	75.9	75.7	71.7	71.4	74.8	76.1	74.1	73.7	75.5	74.2
C-XLM (small) (KD)	74.2	73.2	71.7	72.7	76.4	73.3	75.5	71.6	70.8	73.3

Table 12: Model compression results for the MultiEURLEX task. We report the model's per-language μ-F_1 across the incremental compression steps (S) presented in Section 5.1. We also report the performance for two baselines: (a) fine-tuning the reference pre-trained C-XLM model (FT), and (b) Knowledge Distillation and Vocabulary Pruning. where the student is the reference pre-trained C-XLM (KD).

Model	MultiEURLEX per Language m-F_1									
	EN	FR	DE	NL	IT	ES	PT	PL	EL	m-F_1
Fine-tuned Models										
XLM-R (large)	64.3	67.3	60.3	61.9	58.1	63.6	57.1	60.0	63.5	63.8
XLM-R (base)	56.2	54.7	50.9	53.6	49.5	52.4	52.2	49.8	52.0	53.2
C-XLM (large)	66.8	63.2	63.0	63.9	55.0	67.1	60.6	61.6	70.7	65.4
C-XLM (base)	59.6	58.8	55.1	59.7	54.0	59.1	59.8	57.1	60.3	59.4
C-XLM (small)	55.9	52.7	50.8	55.8	52.6	57.0	55.6	50.8	56.2	54.7
C-XLM (tiny)	50.1	43.7	47.7	42.6	38.3	46.8	47.6	41.9	44.1	46.1
Gradual Compression — Reference C-XLM (tiny)										
Step 1 (VP+KD)	64.3	64.3	60.8	60.8	53.2	66.6	63.3	59.0	56.0	62.9
Step 2 (DP+KD)	58.4	56.9	51.0	55.6	56.4	61.0	55.5	56.2	54.9	58.2
Step 3 (WP+KD)	56.1	62.2	55.5	53.7	53.6	59.7	55.5	54.3	53.9	58.0
Step 4 (GO+Q)	54.8	56.0	54.6	54.2	53.5	58.8	54.0	54.4	51.1	57.0
C-XLM (tiny) (KD)	42.2	41.1	40.9	38.7	41.5	40.9	41.5	41.2	39.4	42.0
Gradual Compression — Reference C-XLM (small)										
Step 1 (VP+KD)	64.3	64.3	60.8	60.8	53.2	66.6	63.3	59.0	56.0	62.9
Step 2 (DP+KD)	58.2	55.8	49.9	56.0	50.2	56.5	53.6	49.8	50.8	54.3
Step 3 (WP+KD)	61.9	60.9	56.2	59.1	53.8	61.8	55.5	61.1	58.4	60.8
Step 4 (GO+Q)	61.5	60.8	57.3	59.8	54.7	61.6	55.4	57.6	58.8	60.4
C-XLM (small) (KD)	56.9	54.3	52.8	56.3	54.3	55.1	55.8	48.6	47.8	54.7

Table 13: Model compression results for the MultiEURLEX task. We report the model's per-language m-F_1 across the incremental compression steps (S) presented in Section 5.1. We also report the performance for two baselines: (a) fine-tuning the reference pre-trained C-XLM model (FT), and (b) Knowledge Distillation and Vocabulary Pruning. where the student is the reference pre-trained C-XLM (KD).

Model	UNFAIR-ToS per Language m-F_1									
	EN		PL		IT		DE		Total	
	MAE	μ-F_1	MAE	μ-F_1	MAE	μ-F_1	MAE	μ-F_1	MAE	μ-F_1
Fine-tuned Models										
XLM-R (large)	0.16	89.3	0.19	87.2	0.14	91.2	0.15	88.3	0.16	89.0
XLM-R (base)	0.14	90.3	0.22	81.7	0.15	88.2	0.17	86.4	0.17	86.6
C-XLM (large)	0.13	90.3	0.17	88.1	0.11	92.2	0.15	88.3	0.14	89.7
C-XLM (base)	0.17	87.4	0.22	83.5	0.13	91.2	0.19	87.4	0.18	87.3
C-XLM (small)	0.17	85.4	0.24	80.7	0.16	90.2	0.21	85.4	85.4	0.20
C-XLM (tiny)	0.27	82.5	0.28	76.1	0.24	79.4	0.30	74.8	78.2	0.27
Gradual Compression — Reference C-XLM (tiny)										
Step 1 (VP+KD)	0.13	92.2	0.21	86.2	0.14	90.2	0.22	85.4	0.18	88.5
Step 2 (DP+KD)	0.23	80.6	0.24	78.0	0.22	84.3	0.27	79.6	0.24	80.6
Step 3 (WP+KD)	0.24	80.6	0.24	80.7	0.23	82.4	0.28	75.7	0.25	79.9
Step 4 (GO+Q)	0.24	80.6	0.24	80.7	0.24	82.4	0.28	75.7	0.25	79.6
C-XLM (tiny) (KD)	0.32	76.7	0.30	75.2	0.30	76.5	0.27	78.6	0.30	76.7
Gradual Compression — Reference C-XLM (small)										
Step 1 (VP+KD)	0.13	92.2	0.21	86.2	0.14	90.2	0.22	85.4	0.18	88.5
Step 2 (DP+KD)	0.20	83.5	0.25	80.7	0.21	84.3	0.21	85.4	0.22	83.5
Step 3 (WP+KD)	0.20	84.5	0.21	84.4	0.22	81.4	0.21	83.5	0.21	83.5
Step 4 (GO+Q)	0.20	84.5	0.20	84.4	0.22	82.4	0.21	83.5	0.21	83.7
C-XLM (small) (KD)	0.25	81.6	0.28	78.0	0.22	83.3	0.26	81.6	0.25	81.1

Table 14: Model compression m-F_1 for the UNFAIR-ToS task. We report the per-language model's μ-F_1 and MAE across the incremental compression steps (S) presented in Section 5.1. We also report the performance for two baselines: (a) fine-tuning the reference pre-trained C-XLM model (FT), and (b) Knowledge Distillation and Vocabulary Pruning. where the student is the reference pre-trained C-XLM (KD).

Model	C-XLM								XLM-R			
	Tiny		**Small**		**Base**		**Large**		**Base**		**Large**	
Corpus Subset	Loss	Acc.	Loss	Acc.	Loss	Acc.	Loss	Acc.	Loss	Acc.	Loss	Acc.
Regulations (EN)	2.46	54.2%	1.62	67.1%	1.11	76.3%	0.97	80.0%	1.36	71.2%	1.07	76.3%
Regulations (EL)	1.89	61.0%	1.24	72.9%	0.85	80.5%	0.74	83.9%	0.93	79.2%	0.68	84.3%
Regulations (DE)	2.52	52.3%	1.60	67.3%	1.02	77.6%	0.84	82.1%	1.20	74.2%	0.89	80.0%
Regulations (FR)	1.84	62.3%	1.16	74.8%	0.75	82.8%	0.65	86.0%	1.01	77.8%	0.77	82.5%
Regulations (ES)	2.01	59.7%	1.31	72.1%	0.88	80.2%	0.78	83.3%	1.19	74.4%	0.93	79.0%
Regulations (NL)	2.43	54.0%	1.54	68.4%	1.00	78.1%	0.85	82.1%	1.25	73.8%	0.91	79.8%
Regulations (IT)	2.06	58.5%	1.32	71.8%	0.88	80.2%	0.77	83.5%	1.16	75.0%	0.88	80.2%
Regulations (PL)	2.23	57.2%	1.44	70.3%	0.94	79.2%	0.75	83.3%	1.08	76.8%	0.80	82.1%
Regulations (PT)	2.20	57.2%	1.43	70.7%	0.98	79.0%	0.87	82.3%	1.23	73.5%	0.94	78.8%
Regulations (All)	2.36	55.1%	1.53	68.9%	1.03	77.7%	0.90	81.4%	1.18	74.6%	0.90	79.8%
Contracts (EN)	1.96	58.0%	1.15	73.7%	0.66	84.2%	0.45	89.1%	1.24	73.0%	0.93	78.8%
Contracts (EL)	2.11	57.2%	1.41	70.0%	0.99	78.0%	0.84	81.8%	1.07	76.8%	0.87	80.9%
Contracts (DE)	2.62	50.0%	1.65	66.2%	1.07	76.5%	0.89	80.7%	1.27	73.0%	1.04	77.4%
Contracts (FR)	1.99	59.9%	1.23	73.9%	0.78	82.5%	0.65	85.9%	1.09	76.8%	0.88	80.7%
Contracts (ES)	2.28	54.7%	1.45	69.2%	0.95	78.6%	0.80	82.5%	1.32	72.4%	1.08	76.4%
Contracts (NL)	2.64	49.7%	1.69	65.2%	1.12	75.5%	0.93	79.9%	1.43	70.6%	1.15	75.5%
Contracts (IT)	2.37	53.4%	1.52	68.4%	1.01	77.8%	0.86	81.7%	1.38	71.5%	1.14	75.9%
Contracts (PL)	2.66	50.2%	1.74	65.1%	1.17	75.0%	0.98	79.4%	1.14	75.3%	0.94	79.5%
Contracts (PT)	2.62	49.8%	1.69	65.4%	1.14	75.4%	0.96	79.6%	1.67	66.8%	1.46	70.2%
Contracts (RU)	1.98	58.2%	1.29	71.6%	0.85	80.5%	0.70	84.6%	1.18	75.4%	0.99	78.8%
Contracts (All)	2.26	56.6%	1.43	71.0%	0.97	79.5%	0.87	82.9%	1.41	71.7%	1.16	76.1%
Overall (EN)	2.28	55.5%	1.44	69.9%	0.93	79.5%	0.76	83.8%	1.32	71.9%	1.02	77.3%
Overall (EL)	2.02	59.4%	1.33	71.7%	0.92	79.5%	0.81	82.9%	1.04	77.5%	0.81	82.1%
Overall (DE)	2.53	52.1%	1.60	67.5%	1.04	77.5%	0.87	81.7%	1.27	73.4%	0.99	78.6%
Overall (FR)	1.92	61.6%	1.20	74.6%	0.78	82.6%	0.69	85.7%	1.09	76.8%	0.86	81.2%
Overall (ES)	2.12	57.9%	1.36	71.2%	0.92	79.6%	0.81	83.0%	1.28	73.1%	1.03	77.6%
Overall (NL)	2.52	52.4%	1.60	67.4%	1.06	77.1%	0.91	81.1%	1.37	72.0%	1.06	77.4%
Overall (IT)	2.20	56.6%	1.40	70.7%	0.94	79.3%	0.83	82.8%	1.30	72.8%	1.03	77.8%
Overall (PL)	2.41	54.3%	1.56	68.3%	1.04	77.6%	0.86	81.7%	1.15	75.9%	0.90	80.5%
Overall (PT)	2.37	54.4%	1.53	68.8%	1.05	77.7%	0.93	81.2%	1.47	69.9%	1.22	74.4%
Overall (RU)	1.98	58.2%	1.29	71.6%	0.85	80.5%	0.70	84.6%	1.18	75.4%	0.99	78.8%
Overall	2.37	54.9%	1.53	69.0%	1.03	77.9%	0.90	81.5%	1.23	74.0%	0.96	79.0%

Table 15: Masked-Language-Models Validation Performance Scores (Cross-Entropy Loss, Accuracy).