Speedy Gonzales: A Collection of Fast Task-Specific Models for Spanish

José Cañete
Department of Computer Science, University of Chile
jose.canete@ug.uchile.cl

Felipe Bravo-Marquez
Department of Computer Science, University of Chile
National Center for Artificial Intelligence (CENIA)
Millennium Institute for Foundational Research on Data (IMFD)
fbravo@dcc.uchile.cl

Abstract

Large language models (LLM) are now a very common and successful path to approach language and retrieval tasks. While these LLM achieve surprisingly good results it is a challenge to use them on more constrained resources. Techniques to compress these LLM into smaller and faster models have emerged for English or Multilingual settings, but it is still a challenge for other languages. In fact, Spanish is the second language with most native speakers but lacks of these kind of resources. In this work, we evaluate all the models publicly available for Spanish on a set of 6 tasks and then, by leveraging on Knowledge Distillation, we present Speedy Gonzales, a collection of inference-efficient task-specific language models based on the ALBERT architecture. All of our models (fine-tuned and distilled) are publicly available on: https://huggingface.co/dccuchile.

1 Introduction

The utilization of learned dense representations of text is nowadays a common and successful approach for different kind of information retrieval (IR) tasks (Yates et al., 2021). These learned representations are usually obtained by training a language model using large collections of texts from the web. Two key aspects to watch to make the most of these models are size and speed of them.

The size of these models has grown overtime and now very large language models (LLM) are common, with models that range from hundred of millions to billions of parameters. These pre-trained models are not only heavy on memory requirements but also on the operations they do on every inference, which is a bottleneck when trying to deploy these models for tasks that are expected to be fast such as question answering or semantic search.

These LLMs are usually trained on English by big technology companies using web-scale datasets and substantial computational resources. Prominent examples include the well-known GPT-3 model (Brown et al., 2020). For languages other than English the available models are typically variants of BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019) or ALBERT (Lan et al., 2020). In the case of Spanish, which is one of the five most spoken languages in the world and the second with most native speakers, the available models range from 5M to 335M of parameters. In Figure 1 we showed how different Spanish pre-trained models compare in terms of model size (number of parameters) and inference speed (MACs).

Despite the remarkable performance of these LLMs across a range of tasks, it remains a challenge to utilize them effectively in computing environments that are constrained by limited resources, such as web or mobile applications.

New techniques to address this problem have emerged for English (Tang et al., 2019; Turc et al., 2019; Sanh et al., 2019; Wang et al., 2020; Jiao et al., 2020) or Multilingual (Jiao et al., 2021) models. These typically leverage on different kinds of Knowledge Distillation (Hinton et al., 2015) to compress the results of a large and performant model into another one which is typically lighter and more inference efficient. For other languages this is still an open challenge, where we lack from this kind of resources.

In this work we try to close this gap with new resources (inference-efficient models) for the Spanish language. Our contributions are the following:

- We perform a comprehensive evaluation of all publicly available Spanish pre-trained models, which are trained on general-domain corpora, by fine-tuning them across six different tasks and eight datasets.

- By selecting the best model on each evaluated dataset, we distilled its knowledge into lighter

Figure 1: The size (number of parameters) and speed (MACs) of every Spanish model evaluated on this work. MACs are measured using a single sequence of length 512, which is the maximum sequence length of all the evaluated models.

ALBERT models, achieving more lighter and inference efficient models, while retaining most of the task performance of the bigger counterparts.

- We make our newly created resource, Speedy Gonzales, consisting of over 140 fine-tuned and distilled models, publicly accessible on the HuggingFace Hub at: `https://huggingface.co/dccuchile`.

2 Related Work

Transformers, introduced by Vaswani et al. (2017) have become the default architecture for text-related tasks. Transformer encoders like BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019) or ALBERT (Lan et al., 2020) are some of the most popular, by its ability to encode complex relations on texts by training on large collections of texts, with the training task consisting of corrupt some parts of a text sequence and train a model to reconstruct the correct sequence.

While models with billions of parameters have become common for English language (Brown et al., 2020), it is not the case for most other languages, which are typically restricted to hundreds of millions of parameters. For Spanish language, which is one of the most spoken languages in the world, the models available follow the BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019) or ALBERT (Lan et al., 2020) architecture and are described in further detail in Section 4.2.

Several ways to compress these models have been proposed through the years. The most common ones are quantization (Gholami et al., 2021), pruning (Blalock et al., 2020) and knowledge distillation (Hinton et al., 2015).

Network quantization compresses the original network by reducing the number of bits required to represent each weight, resulting in a lighter model. In the case of BERT, examples of these kinds of methods are TernaryBERT (Zhang et al., 2020) and BinaryBERT (Bai et al., 2021) where they were able to reduce the weight size to 2 and 1 bit respectively, while maintaining most of the original BERT performance.

The technique of pruning aims to reduce the number of connections (weights) in a neural network, which results in a reduction of the model size and also a very sparse pattern of the weights. Frankle and Carbin (2019) showed that in most feed-forward neural networks it is possible to find a subnetwork that achieves similar or better accuracy.

In Knowledge Distillation (KD) (Hinton et al., 2015) the knowledge learned by a big and strong model, the teacher model, is transferred to a lighter model, the student model, by forcing this student to mimic the teacher. Multiples ways of knowledge distillation have been proposed (Gou et al., 2021).

Tang et al. (2019) uses KD to transfer the knowledge from BERT to lighter RNNs. Turc et al. (2019) proposes pre-training compact BERT models and then using task-specific KD to achieve better results. Sanh et al. (2019) introduces a task-agnostic scheme where KD is used on the pre-training task. Wang et al. (2020) and Jiao et al. (2020) proposed different methods exclusive for

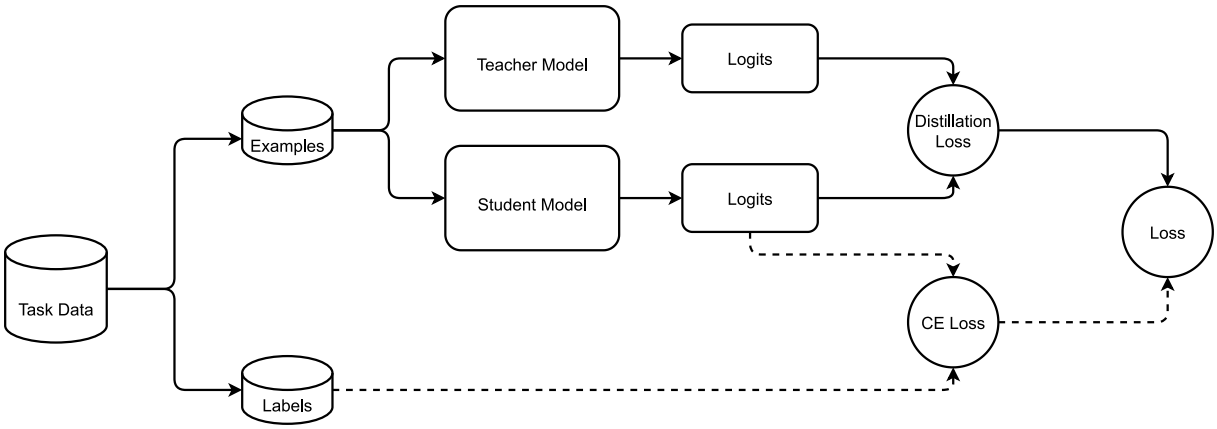

Figure 2: The figure provides a visual representation of the Knowledge Distillation framework applied in this work. In line with common practices, the framework includes both a distillation loss between the teacher and student models and a cross-entropy loss between the gold labels and the student's predictions, as indicated by the dashed line.

Transformers, to directly distill the knowledge from the self-attention layers of the teacher model to the student model.

Our work is similar to Turc et al. (2019) by proposing the use of compact Transformers but we use the ALBERT architecture instead of the BERT one. We also use the idea from Sanh et al. (2019) of reusing the layers of a pre-trained model, instead of random initializing a new one. Differently from that work, that has to choose which layers to reuse, we only adjust the number of layers (and thus, the inference speed) since all the ALBERT layers are shared. Another difference with those two works is that in our work we skip pre-training (or KD on the pre-training task) and directly apply KD on the task-specific phase.

3 Methodology

In pursuit of our goal to have efficient models for Spanish in various tasks, we employ the method of Knowledge Distillation. This method will be further elaborated in the subsequent section.

3.1 Knowledge Distillation

The technique of Knowledge Distillation aims to transfer the knowledge learned from a big and capable model, usually called the teacher model, say M_T, to a more restricted model, called the student model, say M_S. To achieve this objective, we train M_S to imitate M_T. There are multiple ways to imitate M_T (Gou et al., 2021), in this work we use the simple, yet powerful approach, of directly mimic the output of M_T given a input text.

Formally, we define the distillation objective as L_{KD}:

$$L_{KD} = L_O(M_T(x), M_S(x))$$

Where L_O is a loss function that works on the logits of M_T and M_S. The most common choices for this loss are the cross entropy loss, the KL-divergence loss and the mean-squared error loss. In the case of KL-divergence or cross-entropy loss is it a common practice to use soft-targets (Hinton et al., 2015) instead of direct logits, which means to apply a softmax with temperature T (with $T >= 1$) to $M_T(x)$ and $M_S(x)$ in order to produce a soft probability distribution over the classes.

Also, typically we use not only the output of M_T but also the gold labels from the training dataset. The complete loss, accounting these labels can be seen as:

$$L = \alpha L_{CE} + (1 - \alpha) L_{KD}$$

Where L_{CE} is the traditional cross-entropy loss against gold labels and α defines the weight of each loss.

An overview of the entire framework is shown in Figure 2.

3.2 Approach

Our approach has two stages, in the first one, we fine-tune a set of candidate teacher models in a set of tasks of interest. Then, for each task we select the best teacher model (which we define as the model with minimum validation loss among all candidate models) as the teacher model for that task. In stage two, we apply KD using these teachers models and a set of students models.

The complete set of evaluated tasks and possible teacher models is described in Section 4.

3.3 Student Models

For the student models, we rely on the ALBERT (Lan et al., 2020) architecture. This architecture is lighter in terms of parameters because all layers are weight-tied. Specifically, we adopt ALBETO models (Cañete et al., 2022) models, adhering to the ALBERT architecture and exclusively trained for the Spanish language. We considered ALBETO *tiny*, which is the lightest models of all ALBETO models and also, inspired by Sanh et al. (2019) we propose models with less layers (and thus faster) that match the configuration of ALBETO *base*, except on the number of layers. These lighter ALBERTs are then initialized with the weights of ALBETO *base*. These models are noted in the tables as ALBETO *base-n*, where *n* is the number of layers of the model.

3.4 Implementation Details

All our code uses Python and PyTorch (Paszke et al., 2019) as machine learning framework and is publicly available on GitHub[1].

The evaluation of the inference speed of the proposed models is performed through the utilization of the Multiply-Accumulate (MACs) metric, which provides a hardware-agnostic evaluation and is thus considered to be a more robust evaluation criterion. This measurement is conducted using the THOP[2] library, which operates on PyTorch models, to accurately measure MACs. In addition, to provide a more intuitive understanding of the models' performance, actual inference speeds on commonly used hardware configurations are also reported in Section 3.5.

For KD, we first experimented using the three different losses, with different parameters α and T using Optuna (Akiba et al., 2019). These experiments showed that the best results where using $\alpha = 0$ and $T = 1$. With that parameters, while the three different losses works well, KL-divergence was slightly better, so we conducted the rest of the experiments using that configuration.

For both stages of our approach, the only preprocessing applied was tokenization of the input texts according to the subword vocabulary of every model.

For the first stage, which is fine-tuning of the possible teacher models we rely heavily on the HuggingFace Transformers (Wolf et al., 2020) library. For all models and tasks, we run a grid search over the hyperparameters batch size = {16, 32, 64} and epochs = {2, 3, 4}. We experimented with learning rate = {1e-5, 2e-5, 3e-5, 5e-5} for all models except ALBETO *large*, *xlarge*, and *xxlarge*, where we used learning rate = {1e-6, 2e-6, 3e-6, 5e-6}, which are the same hyperparameters used on (Cañete et al., 2022).

For the second stage, which is applying KD, the implementation depended on the task. For text classification tasks we do the KD between the pooled output of both models. For sequence tagging and question answering tasks, we aligned the first token of every word (because the vocabulary of both models is not always the same, which implies that the subword tokenization can result in a different number of tokens) and then we do the KD using the sequence of representations of first tokens for every word in the text between the two models. We note that this approach is not new and is almost the same applied on the original BERT (Devlin et al., 2019) for sequence tagging tasks, that was adapted to work on KD.

For the experiments on this second stage we did a grid search using the hyperparameters: learning rate = {5e-5, 1e-4}, batch sizes = {16, 32, 64} and epochs = 50, we also use early stopping with a tolerance of 10 epochs of no improving.

To accelerate experimentation, we employ a teacher output cache, with its impact on training times discussed in Appendix C.

In Tables 2 and 3 we report results of the models on the test set of each dataset. These models were selected based on the best results on the validation set among the grid search experiments. These models are also the ones publicly available on the HuggingFace Hub.

3.5 Inference Speed on Common Hardware

In our work we measure inference speed in terms of Multiply-Accumulate (MAC) operations. This metric is advantageous as it is agnostic to hardware variations. However, it can be useful to also report the actual inference speed of models on common hardware, as this can provide a more intuitive understanding of their performance.

Table 1 presents the average number of inferences per second that can be achieved on two different hardware platforms, a CPU with an In-

[1] https://github.com/dccuchile/speedy-gonzales
[2] https://github.com/Lyken17/pytorch-OpCounter

Model	Inferences per second	
	CPU	**GPU**
Fine-tuning		
BETO *uncased*	3.96	107.19
BETO *cased*	4.26	109.02
DistilBETO	9.12	217.40
ALBETO *tiny*	32.53	539.61
ALBETO *base*	4.50	108.62
ALBETO *large*	1.29	33.62
ALBETO *xlarge*	0.35	11.72
ALBETO *xxlarge*	0.14	6.60
BERTIN	3.99	109.39
RoBERTa BNE *base*	3.82	107.77
RoBERTa BNE *large*	1.18	33.65
Task-specific Knowledge Distillation		
ALBETO *tiny*	32.53	539.61
ALBETO *base-2*	31.08	625.30
ALBETO *base-4*	15.16	319.32
ALBETO *base-6*	10.45	213.53
ALBETO *base-8*	6.82	160.66
ALBETO *base-10*	6.01	128.38

Table 1: The number of inferences per second of each model on two different hardware settings, CPU and GPU.

tel Core i7-11700K and a GPU with a NVIDIA GeForce RTX 3090. To account for variance in the measurements, we first conducted 10 warm-up inferences followed by 100 real measures for each model. We then applied an aggressive outlier filtering method based on the modified Z-Score (Iglewicz and Hoaglin, 1993) with a threshold of 0.75, which resulted in the removal of approximately 40-45% of the measures. The remaining 55-60% of the measures were used to calculate, with very low variance, the average inference speed (in milliseconds) and the number of inferences that could be performed in one second, which serves as a clearer illustration of the model's inference speed.

It is worth noting that the difference in speed between the larger models and the proposed models trained using task-specific KD is substantial. Specifically, on the CPU setting, which is representative of popular serverless platforms used in industry, the best model found in this study in terms of task performance, ALBETO *xxlarge*, would take several seconds for a single inference, making it unsuitable for real-time user-facing applications. On the other hand, if we consider our proposed faster models, we can observe that ALBETO *base-6* is capable of executing more than 10 inferences per second, which is a much more acceptable latency for a real-time application.

4 Evaluating Spanish Pre-trained Language Models

In order to achieve our goal of have efficient models for Spanish in a variety of tasks we first define a set of tasks to evaluate those models. These tasks are the same evaluated by Cañete et al. (2022) and are described in Section 4.1. We then define a set of possible teacher models, in particular, we wanted to try every model that was pre-trained on general domain Spanish text and is publicly available, therefore we exclude RigoBERTa (Serrano et al., 2022), which is a DeBERTa (He et al., 2021) model for Spanish that is not public and RoBERTuito (Pérez et al., 2022) which is a RoBERTa-like model for Spanish that was trained on Twitter datasets and should be better suited for social media related tasks. All considered models are described in Section 4.2. After evaluating all models on each task, we selected the model with lowest validation loss as the teacher model for the task. The list of selected models can be found in Appendix A.

4.1 Tasks and Data

4.1.1 Document Classification

The task of document classification consists on the assignment of an entire document to a category according to its semantic meaning. For our evaluation we are using the Spanish portion of MLDoc (Schwenk and Li, 2018) which is a multilingual dataset for document classification in eight languages. MLDoc is based on the Reuters Corpus (Lewis et al., 2004) and has four different categories for its documents, which are: Corporate/Industrial, Economics, Government/Social and Markets.

4.1.2 Paraphrase Identification

On Paraphrase Identification we aim to assess whether two sentences share the same semantic meaning. To evaluate our models in this task we are using the Spanish subset of PAWS-X (Yang et al., 2019). This dataset can be seen as a translation to six different languages of the PAWS (Zhang et al., 2019) dataset, where the train set is machine translated and the validation and test sets were translated professionally by humans.

Model	Text Classification (Accuracy)			Sequence Tagging (F1 Score)		Question Answering (F1 Score / Exact Match)		
	MLDoc	**PAWS-X**	**XNLI**	**POS**	**NER**	**MLQA**	**SQAC**	**TAR / XQuAD**
Fine-tuning								
BETO *uncased*	96.38	84.25	77.76	97.81	80.85	64.12 / 40.83	72.22 / 53.45	74.81 / 54.62
BETO *cased*	96.65	89.80	81.98	98.95	**87.14**	67.65 / 43.38	78.65 / 60.94	77.81 / 56.97
DistilBETO	96.35	75.80	76.59	97.67	78.13	57.97 / 35.50	64.41 / 45.34	66.97 / 46.55
ALBETO *tiny*	95.82	80.20	73.43	97.34	75.42	51.84 / 28.28	59.28 / 39.16	66.43 / 45.71
ALBETO *base*	96.07	87.95	79.88	98.21	82.89	66.12 / 41.10	77.71 / 59.84	77.18 / 57.05
ALBETO *large*	92.22	86.05	78.94	97.98	82.36	65.56 / 40.98	76.36 / 56.54	76.72 / 56.21
ALBETO *xlarge*	95.70	89.05	81.68	98.20	81.42	68.26 / 43.76	78.64 / 59.26	**80.15 / 59.66**
ALBETO *xxlarge*	96.85	89.85	**82.42**	98.43	83.06	**70.17 / 45.99**	**81.49** / 62.67	79.13 / 58.40
BERTIN	96.47	88.65	80.50	**99.02**	85.66	66.06 / 42.16	78.42 / 60.05	77.05 / 57.14
RoBERTa BNE *base*	96.82	89.90	81.12	99.00	86.80	67.31 / 44.50	80.53 / **62.72**	77.16 / 55.46
RoBERTa BNE *large*	**97.00**	**90.00**	51.62	61.83	21.47	67.69 / 44.88	80.41 / 62.14	77.34 / 56.97
Task-specific Knowledge Distillation								
ALBETO *tiny*	96.40	85.05	75.99	97.36	72.51	54.17 / 32.22	63.03 / 43.35	67.47 / 46.13
ALBETO *base-2*	96.20	76.75	73.65	97.17	69.69	48.62 / 26.17	58.40 / 39.00	63.41 / 42.35
ALBETO *base-4*	96.35	86.40	78.68	97.60	74.58	62.19 / 38.28	71.41 / 52.87	73.31 / 52.43
ALBETO *base-6*	96.40	88.45	81.66	97.82	78.41	66.35 / 42.01	76.99 / 59.00	75.59 / **56.72**
ALBETO *base-8*	96.70	89.75	**82.55**	97.96	80.23	67.39 / 42.94	77.79 / 59.63	77.89 / **56.72**
ALBETO *base-10*	**96.88**	**89.95**	82.26	**98.00**	**81.10**	**68.29 / 44.29**	**79.89 / 62.04**	**78.21** / 56.21

Table 2: Results of every evaluated model on the test set of each task. On Text Classification datasets (MLDoc, PAWS-X, XNLI) we use Accuracy as metric. For POS and NER, which are Sequence Tagging tasks, we report the F1 Score. On Question Answering, we report two metrics, noted as F1 Score / Exact Match.

Model	Parameters	Speedup	Score
Fine-tuning			
BETO *uncased*	110M	1.00x	81.02
BETO *cased*	110M	1.00x	84.82
DistilBETO	67M	2.00x	76.73
ALBETO *tiny*	**5M**	**18.05x**	74.97
ALBETO *base*	12M	0.99x	83.25
ALBETO *large*	18M	0.28x	82.02
ALBETO *xlarge*	59M	0.07x	84.13
ALBETO *xxlarge*	223M	0.03x	**85.17**
BERTIN	125M	1.00x	83.97
RoBERTa BNE *base*	125M	1.00x	84.83
RoBERTa BNE *large*	355M	0.28x	68.42
Task-specific Knowledge Distillation			
ALBETO *tiny*	**5M**	**18.05x**	76.49
ALBETO *base-2*	12M	5.96x	72.98
ALBETO *base-4*	12M	2.99x	80.06
ALBETO *base-6*	12M	1.99x	82.70
ALBETO *base-8*	12M	1.49x	83.78
ALBETO *base-10*	12M	1.19x	**84.32**

Table 3: The summary of results of every evaluated model in terms of parameters, inference speedup and overall score across tasks. The speedup is relative to BETO models. The score column shows the average of the metrics on all tasks.

4.1.3 Natural Language Inference

In the task of Natural Language Inference we are given two sentences, an "hypothesis" and a "premise", and our task is to determine if one entails the other one, contradicts it or is neutral to it. For this task we use the Spanish subset of XNLI (Conneau et al., 2018), which, very similarly to PAWS-X, offers a machine translated train set from MultiNLI (Williams et al., 2018) and professionally translated validation and test sets to 15 languages.

4.1.4 Part of Speech Tagging

The objective of the task of Part of Speech Tagging is to label words within a sentence according to its corresponding syntactic categories. There are different categories of parts of speech, for example, nouns, verbs, adjectives, adverbs, pronouns, etc. In this task the dataset used was AnCora (Taulé et al., 2008) which is included on the Spanish part of Universal Dependencies (de Marneffe et al., 2021) Treebank.

4.1.5 Named Entity Recognition

Named Entity Recognition is a sequence labeling task in which the goal is to classify entities within a text with their corresponding type. These types are usually names of people, places, organizations or miscellaneous. These entities can be formed by more than one word, that is why the datasets typi-

cally adopt the BIO annotation, which means for a word that it can be the beggining (B) of a entity, inside (I) a entity or out (O) of it. For this task the dataset used as evaluation is from the shared task of CoNLL-2002 (Tjong Kim Sang, 2002), we use the Spanish subset of it.

4.1.6 Question Answering

There are different types of Question Answering tasks. In this evaluation our focus is Extractive Question Answering, that is, given a context text and question about that context, point out the span of words that fully answers the question. On this task we considered four different datasets, which are, MLQA (Lewis et al., 2020), SQAC (Gutiérrez-Fandiño et al., 2022), TAR (Carrino et al., 2020) and XQuAD (Artetxe et al., 2020). MLQA is a multilingual dataset created by using English QA instances and then professionally translated them to six different languages, from these they provide a validation and a test set, but they also provide a machine translated version of SQuAD v1.1 (Rajpurkar et al., 2016) as train set to each of the languages, we use the Spanish subsets of it. TAR offers a different machine translated dataset from SQuAD v1.1 to Spanish. XQuAD provides a test set obtained from SQuAD v1.1 and professionally translated to 11 different languages. Following the setup by (Cañete et al., 2020) we pair the train and validation sets from TAR and the Spanish test set from XQuAD as a single evaluation dataset. Finally, SQAC is the only dataset evaluated that was built exclusively for Spanish.

4.2 Models

4.2.1 BETO

BETO (Cañete et al., 2020) is the first Transformer encoder pre-trained exclusively on Spanish corpora. It is BERT-base sized model that has two versions available, *uncased* and *cased*. They have an approximate of 110M parameters and each have a vocabulary of 31K BPE (Sennrich et al., 2016) subwords which was constructed using SentencePiece (Kudo and Richardson, 2018). Both models were trained for 2M optimization steps on the SUC (Cañete, 2019) dataset.

4.2.2 ALBETO

ALBETO (Cañete et al., 2022) is a series of ALBERT (Lan et al., 2020) models for Spanish. There are 5 different sizes, that range from 5M to 223M parameters, which are *tiny*, *base*, *large*, *xlarge* and *xxlarge*. The *tiny* model is similar to the one trained on Chinese [3], the rest follow closely the configurations trained on the original ALBERT work. They share a vocabulary of 31K lowercase BPE (Sennrich et al., 2016) subwords created using SentencePiece (Kudo and Richardson, 2018). All ALBETO models were trained on SUC (Cañete, 2019).

4.2.3 DistilBETO

DistilBETO (Cañete et al., 2022) is a lighter Transformer encoder based on the weights of BETO and further pre-trained using the knowledge distillation technique presented by (Sanh et al., 2019) on DistilBERT. It has 67M parameters and uses the same lowercase vocabulary from BETO *uncased*.

4.2.4 RoBERTa-BNE

RoBERTa-BNE (Gutiérrez-Fandiño et al., 2022) are two different sized RoBERTa (Liu et al., 2019) models trained on Spanish using the National Library of Spain (BNE) (Gutiérrez-Fandiño et al., 2022) corpus which is also the larger Spanish corpus of this type to this date. The *base* model has 125M parameters while the *large* version has 355M. Both version share a vocabulary of 50K BPE (Sennrich et al., 2016) subwords.

4.2.5 BERTIN

BERTIN (de la Rosa et al., 2022) is a RoBERTa-base model trained on the Spanish portion of the mC4 (Raffel et al., 2020) dataset. It has the same size, configuration and vocabulary of the RoBERTa-BNE *base* model.

5 Results

Table 2 presents the results of each model across all evaluated tasks. A general observation is that there are two distinct behaviors among the tasks. Firstly, there is minimal variation in performance between smaller and larger models in certain tasks, as evidenced by the comparable high scores achieved by all models in the MLDoc and POS tasks. It is hypothesized that these tasks are relatively simple, and as a result, the utilization of larger models results in overparameterization.

Secondly, there are tasks where there is a notable difference in performance between smaller and bigger models. This is evident in tasks such as Paraphrase Identification (PAWS-X), Natural Language Inference (XNLI), Named Entity Recognition (NER) and Question Answering (MLQA,

[3] https://github.com/ckiplab/ckip-transformers

SQAC, TAR/XQuAD), where the larger models tend to outperform the smaller models. This suggests that these tasks are more complex and require a greater model capacity. Overall, the results of this evaluation demonstrate the importance of considering the appropriate model size for a given task, as overparameterization can lead to suboptimal inference performance.

5.1 Text Classification

In our experiments on text classification tasks, we observed that models with a depth of 8 or more layers exhibit performance comparable to the best larger models, while also demonstrating significant improvements in inference time. Specifically, for the XNLI dataset, we found that the ALBETO *base-8* model outperforms all other models evaluated in our study.

5.2 Sequence Tagging

On NER we observe a significant difference between our faster models and the *cased* models (BETO, BERTIN, RoBERTa-BNE), especially with BETO *cased*, which was the best model on the task. Furthermore, we observe a difference of almost 4.1 percentual difference (pd) between ALBETO *xxlarge*, and BETO *cased*, even though ALBETO *xxlarge* is one of the largest models in the fine-tuning setting. Additionally, we find a difference of almost 6.3 pd between the *cased* and *uncased* versions of BETO. Based on these observations, we posit that the difference in performance between *cased* and *uncased* models can be attributed to the additional hints provided by capitalization for solving the NER task. Specifically, the names of persons, organizations, and places typically begin with a capital letter. Furthermore, our results from models trained using knowledge distillation (KD) suggest that this hint is not easily replicable in an uncased model.

5.3 Question Answering

The performance on Question Answering datasets, as indicated in the final three columns of the table, follows a pattern similar to that observed in text classification tasks. The larger models, specially ALBETO *xxlarge* and *xlarge*, exhibit higher performance, while our proposed models featuring 8 or more layers present results similar to those of the base-sized models.

5.4 Discussion and Summary

It should be noted that some models performed significantly worse than the others. Specifically, the utilization of RoBERTa-BNE *large* on XNLI, POS, and NER tasks produced subpar results. This deviation from the performance of the same model on other tasks, as well as the results reported by Gutiérrez-Fandiño et al. (2022), suggests that RoBERTa-BNE *large* may be particularly sensitive to hyperparameter selection and may benefit from additional hyperparameter tuning.

Our results show a general progression in performance of our proposed models as the number of layers increases. A clear trade-off between task performance and inference speed is observed, with a more pronounced effect in text classification and question answering tasks, and a weaker effect in sequence tagging. Additionally, at equal inference speed, our models trained with task-specific distillation exhibit improved performance compared to DistilBETO, which was trained with task-agnostic distillation, despite having significantly fewer parameters.

A similar effect can be observed when comparing ALBETO *base-{8-10}* to the original 12-layer ALBETO *base* fine-tuned using standard techniques, the former exhibits improved performance. This underscores the vital role of task-specific knowledge distillation in obtaining improved performance for these faster models. Additional experiments comparing straightforward fine-tuning and the application of knowledge distillation on these more compact and faster models are presented in Appendix B.

Table 3 summarizes our findings. Following the methodology of GLUE (Wang et al., 2018), we compute a global score that encompasses all tasks, which is displayed in the third column. The score is the simple mean of the individual task results. In the instance of Question Answering, which provides two metrics, we opted for the F1 Score as the representative score for the task. The ALBETO *xxlarge* model achieved the best overall performance, although it was also the slowest and had the second largest number of parameters. With a mere 0.35 performance drop from the top model, the RoBERTa BNE *base* and BETO *cased* models exhibited comparable results. The ALBETO *base-10*, exhibiting a 19% improvement in speed compared to BETO models, is our strongest proposed model with a difference of approximately 0.5

performance drop from the aforementioned models. Our remaining models display varying degrees of improved inference speed, at the expense of slight reductions in task performance.

6 Conclusion and Future Work

In this work, we introduce Speedy Gonzales, a novel resource for the Spanish NLP and IR communities comprising a collection of computationally efficient language models trained on six tasks and eight datasets. By applying the Knowledge Distillation technique, our models achieve comparable performance to state-of-the-art models, while showing faster inference speeds.

The full collection of models, including our proposed models and all the teacher models fine-tuned on the tasks considered, are made publicly available for further research.

We believe that the availability of these models and the expansion of the Knowledge Distillation method to additional tasks will drive the widespread utilization of large language models in the Spanish speaking community, particularly for individuals and organizations seeking to tackle crucial information retrieval challenges, such as question answering, text similarity and semantic search, in both academic and industrial settings.

Potential directions for future research include exploring the use of multiple teachers in the distillation process and developing metrics to formally evaluate the balance between inference speed and task performance.

Acknowledgements

This work was supported by ANID Millennium Science Initiative Program Code ICN17_002 and the National Center for Artificial Intelligence CENIA FB210017, Basal ANID.

References

Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. 2019. Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2019, Anchorage, AK, USA, August 4-8, 2019*, pages 2623–2631. ACM.

Mikel Artetxe, Sebastian Ruder, and Dani Yogatama. 2020. On the cross-lingual transferability of monolingual representations. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4623–4637, Online. Association for Computational Linguistics.

Haoli Bai, Wei Zhang, Lu Hou, Lifeng Shang, Jin Jin, Xin Jiang, Qun Liu, Michael Lyu, and Irwin King. 2021. BinaryBERT: Pushing the limit of BERT quantization. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4334–4348, Online. Association for Computational Linguistics.

Davis Blalock, Jose Javier Gonzalez Ortiz, Jonathan Frankle, and John Guttag. 2020. What is the state of neural network pruning? In *Proceedings of Machine Learning and Systems*, volume 2, pages 129–146.

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.

José Cañete, Sebastian Donoso, Felipe Bravo-Marquez, Andrés Carvallo, and Vladimir Araujo. 2022. ALBETO and DistilBETO: Lightweight Spanish language models. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 4291–4298, Marseille, France. European Language Resources Association.

Casimiro Pio Carrino, Marta R. Costa-jussà, and José A. R. Fonollosa. 2020. Automatic Spanish translation of SQuAD dataset for multi-lingual question answering. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 5515–5523, Marseille, France. European Language Resources Association.

José Cañete. 2019. Compilation of Large Spanish Unannotated Corpora.

José Cañete, Gabriel Chaperon, Rodrigo Fuentes, Jou-Hui Ho, Hojin Kang, and Jorge Pérez. 2020. Spanish pre-trained bert model and evaluation data. In *PML4DC at ICLR 2020*.

Alexis Conneau, Ruty Rinott, Guillaume Lample, Adina Williams, Samuel Bowman, Holger Schwenk, and Veselin Stoyanov. 2018. XNLI: Evaluating cross-lingual sentence representations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2475–2485, Brussels, Belgium. Association for Computational Linguistics.

Javier de la Rosa, Eduardo G Ponferrada, Paulo Villegas, Pablo González de Prado Salas, Manu Romero, and Maria Grandury. 2022. Bertin: Efficient pre-training of a spanish language model using perplexity sampling. *Procesamiento del Lenguaje Natural*, 68(0):13–23.

Marie-Catherine de Marneffe, Christopher D. Manning, Joakim Nivre, and Daniel Zeman. 2021. Universal Dependencies. *Computational Linguistics*, 47(2):255–308.

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.

Jonathan Frankle and Michael Carbin. 2019. The lottery ticket hypothesis: Finding sparse, trainable neural networks. In *International Conference on Learning Representations*.

Amir Gholami, Sehoon Kim, Zhen Dong, Zhewei Yao, Michael W. Mahoney, and Kurt Keutzer. 2021. A survey of quantization methods for efficient neural network inference. *CoRR*, abs/2103.13630.

Jianping Gou, Baosheng Yu, Stephen J. Maybank, and Dacheng Tao. 2021. Knowledge distillation: A survey. *Int. J. Comput. Vision*, 129(6):1789–1819.

Asier Gutiérrez-Fandiño, Jordi Armengol Estapé, Marc Pàmies, Joan Llop Palao, Joaquin Silveira Ocampo, Casimiro Pio Carrino, Carme Armentano Oller, Carlos Rodriguez Penagos, Aitor Gonzalez Agirre, and Marta Villegas. 2022. Maria: Spanish language models. *Procesamiento del Lenguaje Natural*, 68.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2021. Deberta: decoding-enhanced bert with disentangled attention. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Geoffrey Hinton, Oriol Vinyals, Jeff Dean, et al. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2(7).

Boris Iglewicz and David C Hoaglin. 1993. *Volume 16: how to detect and handle outliers*. Quality Press.

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.

Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. 2021. Lightmbert: A simple yet effective method for multilingual bert distillation. *arXiv preprint arXiv:2103.06418*.

Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, Brussels, Belgium. Association for Computational Linguistics.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. ALBERT: A lite BERT for self-supervised learning of language representations. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

David D. Lewis, Yiming Yang, Tony G. Rose, and Fan Li. 2004. RCV1: A new benchmark collection for text categorization research. *J. Mach. Learn. Res.*, 5:361–397.

Patrick Lewis, Barlas Oguz, Ruty Rinott, Sebastian Riedel, and Holger Schwenk. 2020. MLQA: Evaluating cross-lingual extractive question answering. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7315–7330, Online. Association for Computational Linguistics.

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

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Juan Manuel Pérez, Damián Ariel Furman, Laura Alonso Alemany, and Franco M. Luque. 2022. RoBERTuito: a pre-trained language model for social media text in Spanish. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 7235–7243, Marseille, France. European Language Resources Association.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

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

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of BERT: smaller, faster, cheaper and lighter. *CoRR*, abs/1910.01108.

Holger Schwenk and Xian Li. 2018. A corpus for multilingual document classification in eight languages. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

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.

Alejandro Vaca Serrano, Guillem García Subies, Helena Montoro Zamorano, Nuria Aldama Garcia, Doaa Samy, David Betancur Sánchez, Antonio Moreno-Sandoval, Marta Guerrero Nieto, and Álvaro Barbero Jiménez. 2022. Rigoberta: A state-of-the-art language model for spanish. *CoRR*, abs/2205.10233.

Raphael Tang, Yao Lu, Linqing Liu, Lili Mou, Olga Vechtomova, and Jimmy Lin. 2019. Distilling task-specific knowledge from BERT into simple neural networks. *CoRR*, abs/1903.12136.

Mariona Taulé, M. Antònia Martí, and Marta Recasens. 2008. AnCora: Multilevel annotated corpora for Catalan and Spanish. In *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08)*, Marrakech, Morocco. European Language Resources Association (ELRA).

Erik F. Tjong Kim Sang. 2002. Introduction to the CoNLL-2002 shared task: Language-independent named entity recognition. In *COLING-02: The 6th Conference on Natural Language Learning 2002 (CoNLL-2002)*.

Iulia Turc, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Well-read students learn better: On the importance of pre-training compact models. *arXiv preprint arXiv:1908.08962v2*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.

Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. 2020. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS'20, Red Hook, NY, USA. Curran Associates Inc.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122, New Orleans, Louisiana. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Yinfei Yang, Yuan Zhang, Chris Tar, and Jason Baldridge. 2019. PAWS-X: A cross-lingual adversarial dataset for paraphrase identification. 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 3687–3692, Hong Kong, China. Association for Computational Linguistics.

Andrew Yates, Rodrigo Nogueira, and Jimmy Lin. 2021. Pretrained transformers for text ranking: BERT and beyond. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: Tutorials*, pages 1–4, Online. Association for Computational Linguistics.

Wei Zhang, Lu Hou, Yichun Yin, Lifeng Shang, Xiao Chen, Xin Jiang, and Qun Liu. 2020. TernaryBERT: Distillation-aware ultra-low bit BERT. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 509–521, Online. Association for Computational Linguistics.

Yuan Zhang, Jason Baldridge, and Luheng He. 2019. PAWS: Paraphrase adversaries from word scrambling.

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 1298–1308, Minneapolis, Minnesota. Association for Computational Linguistics.

A Selected Teacher Models

Table 4 presents the teacher models selected for each task. The selection process is based on the lowest validation loss achieved among the candidate teacher models that were fine-tuned for each task.

Dataset	Teacher Model
MLDoc	RoBERTa BNE *large*
PAWS-X	ALBETO *xxlarge*
XNLI	ALBETO *xxlarge*
POS	RoBERTa BNE *base*
NER	RoBERTa BNE *base*
MLQA	ALBETO *xxlarge*
SQAC	ALBETO *xxlarge*
TAR / XQuAD	ALBETO *xxlarge*

Table 4: The teacher models selected for each task.

B Importance of Knowledge Distillation

In addition to other experiments, we conducted ablation experiments to evaluate the contribution of Task-Specific Knowledge Distillation to the results of our faster models based on ALBETO.

Tables 5, 6, and 7 compare the performance of each of our proposed models under two training settings: regular fine-tuning (FT) and task-specific knowledge distillation (KD). For fine-tuning and KD we followed the settings described in Section 3.4.

Overall, our results indicate that training using KD generally yields better results than simple fine-tuning, except for sequence tagging tasks (POS, NER), where the results are mixed.

Table 5 presents the results of text classification tasks, where we observe that KD outperforms fine-tuning. In MLDoc, which is hypothesized as an easier task, the performance is similar for both training schemes and different models. However, in PAWS-X and XNLI, we observe a significant difference between the fine-tuning and KD training schemes.

Table 6 presents the results for sequence tagging tasks, where the performance of models under the KD and fine-tuning settings are mixed. Unlike other types of tasks, where the KD training method is the clear winner, the results here vary. In the case of NER, faster models perform better under the fine-tuning setting, while those with larger compute requirements perform better under the KD setting.

Finally, Table 7 presents the results for question answering, where we observe that models trained using KD generally exhibit better performance than those trained using simple fine-tuning, with a significant difference of around 3-4 percentage points, depending on the model and dataset.

In summary, our results underscore the significance of KD, particularly for harder tasks where the effect is more pronounced, allowing for lighter and faster models to achieve better task performance.

C Effect of Caching Teacher Outputs During Training

A significant challenge in our experimental study is the use of large and costly language models as teacher models for our faster and lighter models. Despite this, as discussed in Appendix B, the importance of these teacher models is essential for achieving better results with our proposed models.

Thus, the use of these teacher models poses challenges in terms of experimentation, particularly when working with restricted budgets, as is often the case in research outside big tech companies. To mitigate this issue, we implement a cache for the outputs of the teacher model, which allows us to train and experiment more efficiently.

The idea behind this approach is straightforward: since the teacher model is fixed during training, its outputs on an input x remain unchanged during different epochs, allowing us to compute them once and reuse them in subsequent epochs.

Formally, suppose F_t and F_s represent the computational cost of the forward pass for the teacher and student models, respectively, on an entire dataset, and E is the number of epochs used to train our proposed models. By caching the teacher's output, the total cost of computing the forward pass reduces from $O(E \cdot (F_t + F_s))$ to $O(F_t + E \cdot F_s)$.

It is worth noting that typically $F_t >> F_s$, and the number of epochs used in knowledge distillation is often higher than that used in simple fine-tuning. To illustrate, our fine-tuning experiments employ between 2 and 4 epochs, while our knowledge distillation experiments use a maximum of 50 epochs.

Model	MLDoc		PAWS-X		XNLI	
	FT	KD	FT	KD	FT	KD
ALBETO *tiny*	95.82	**96.40**	80.20	**85.05**	73.43	**75.99**
ALBETO *base-2*	94.67	**96.20**	73.45	**76.75**	72.08	**73.65**
ALBETO *base-4*	95.88	**96.35**	82.90	**86.40**	75.83	**78.68**
ALBETO *base-6*	95.88	**96.40**	85.20	**88.45**	78.42	**81.66**
ALBETO *base-8*	95.82	**96.70**	87.30	**89.75**	79.44	**82.55**
ALBETO *base-10*	95.65	**96.88**	88.80	**89.95**	79.62	**82.26**

Table 5: Comparison of the performance of our proposed models on text classification tasks on two settings: fine-tuning and task-specific knowledge distillation.

Model	POS		NER	
	FT	KD	FT	KD
ALBETO *tiny*	97.34	**97.36**	**75.42**	72.51
ALBETO *base-2*	**97.46**	97.17	**71.70**	69.69
ALBETO *base-4*	**97.87**	97.60	**76.18**	74.58
ALBETO *base-6*	**98.03**	97.82	78.10	**78.41**
ALBETO *base-8*	**98.18**	97.96	79.46	**80.23**
ALBETO *base-10*	**98.17**	98.00	80.46	**81.10**

Table 6: Comparison of the performance of our proposed models on sequence tagging tasks on two settings: fine-tuning and task-specific knowledge distillation.

To evaluate the impact of our cache implementation, we compare the training times of our proposed models on the XNLI dataset, which is the largest dataset considered in this study, for only 5 epochs (1/10 of the epochs used in our primary experiments) when using the cache and when not using it. Table 8 reports the results of this experiment, presenting the mean (noted as M) and standard deviation (noted as SD) over three runs. As expected, the use of the cache reduces the training time significantly, with results indicating that training time is approximately 1/4 of the time required to train without a cache. This reduction in training time is expected since the forward pass of the teacher model is the most costly operation and is computed only in the first epoch and then retrieved in the next 4 epochs. Furthermore, this difference will increase as the number of epochs increases.

In conclusion, while our cache implementation is a simple engineering trick, it has a significant impact on our experimentation phase in terms of training time and required compute.

Model	MLQA		SQAC		TAR, XQuAD	
	FT	KD	FT	KD	FT	KD
ALBETO *tiny*	51.84 / 28.28	**54.17 / 32.22**	59.28 / 39.16	**63.03 / 43.35**	66.43 / 45.71	**67.47 / 46.13**
ALBETO *base-2*	45.97 / 23.60	**48.62 / 26.17**	53.32 / 34.34	**58.40 / 39.00**	61.82 / 40.67	**63.41 / 42.35**
ALBETO *base-4*	59.99 / 35.69	**62.19 / 38.28**	65.66 / 45.54	**71.41 / 52.87**	68.91 / 49.07	**73.31 / 52.43**
ALBETO *base-6*	63.75 / 38.58	**66.35 / 42.01**	72.22 / 53.61	**76.99 / 59.00**	74.33 / 52.68	**75.59 / 54.95**
ALBETO *base-8*	64.99 / 40.58	**67.39 / 42.94**	75.22 / 56.43	**77.79 / 59.63**	75.47 / 54.11	**77.89 / 56.72**
ALBETO *base-10*	66.29 / 41.69	**68.29 / 44.29**	77.14 / 59.21	**79.89 / 62.04**	77.06 / **56.47**	**78.21** / 56.21

Table 7: Comparison of the performance of our proposed models on question answering on two settings: fine-tuning and task-specific knowledge distillation.

Model	Training Time (hours)			
	Cache		No Cache	
	M	SD	M	SD
ALBETO *tiny*	3.8	3.1×10^{-2}	16.2	3.1×10^{-3}
ALBETO *base-2*	3.8	1.6×10^{-3}	16.3	3.6×10^{-3}
ALBETO *base-4*	4.2	3.3×10^{-4}	16.6	2.6×10^{-3}
ALBETO *base-6*	4.5	1.5×10^{-3}	17.0	1.5×10^{-3}
ALBETO *base-8*	4.8	1.9×10^{-4}	17.3	5.8×10^{-3}
ALBETO *base-10*	5.3	9.6×10^{-3}	17.6	5.6×10^{-3}

Table 8: Training times when using teacher cache vs not using it. Table report the mean (M) and standard deviation (SD) over three runs.