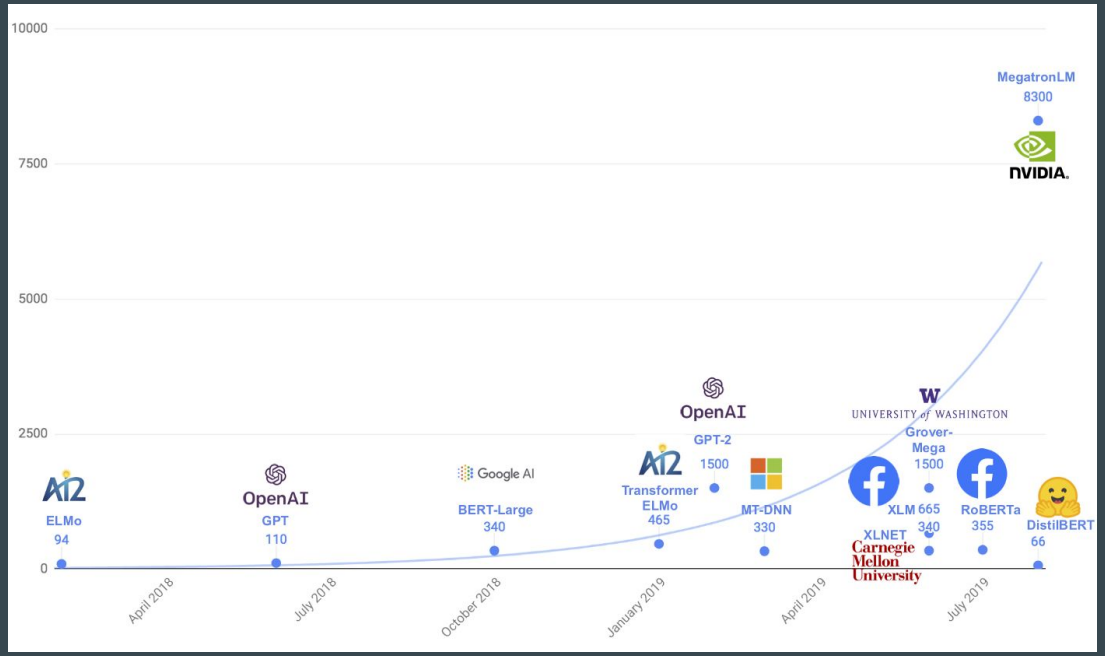
Knowledge Distillation

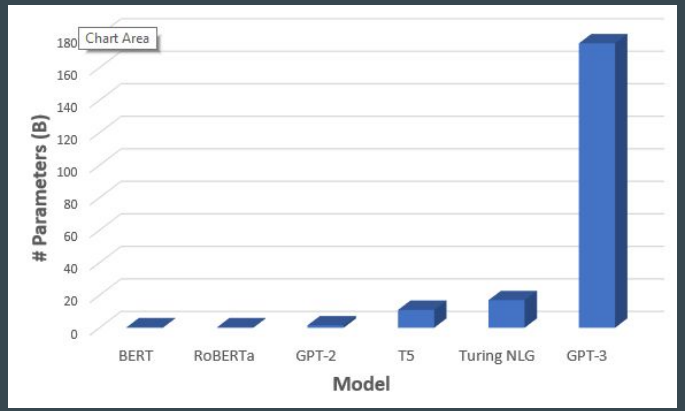
Paper presentation - Shrey Pandit

Why do we need Model compression?

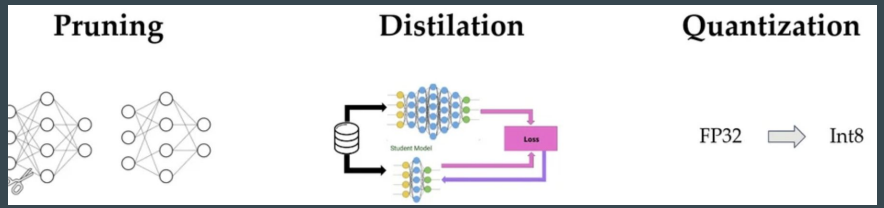


20B-parameter Alexa model sets new marks in few-shot learning

03-Aug-2022 — The lights service doesn't have to know anything about the reminder service. The reminder service needs "one line of code (tm)" to go "After ...



Complex models -> More memory and slower inference time (not ideal for deployment)



Content:

- **MiniLM** : Deep Self-Attention Distillation for Task-Agnostic Compression of Pre-Trained Transformers

- **Load What You Need** : Smaller Versions of Multilingual BERT

- Our Results

MiniLM

Main Contributions -

1) Proposes Self Attention distillation which makes the student model architecture agnostic
   a) As opposed to DistilBERT that has half the number of layers, or,
   b) MobileBERT that needs to have the same number of layers as teacher

2) Scaled dot product that helps to closely mimic the Value in attention without the need to have an extra parameter dimension

3) Introduction of teacher assistant for distilling knowledge from large teacher to smaller student

MiniLM: Last Attention layer

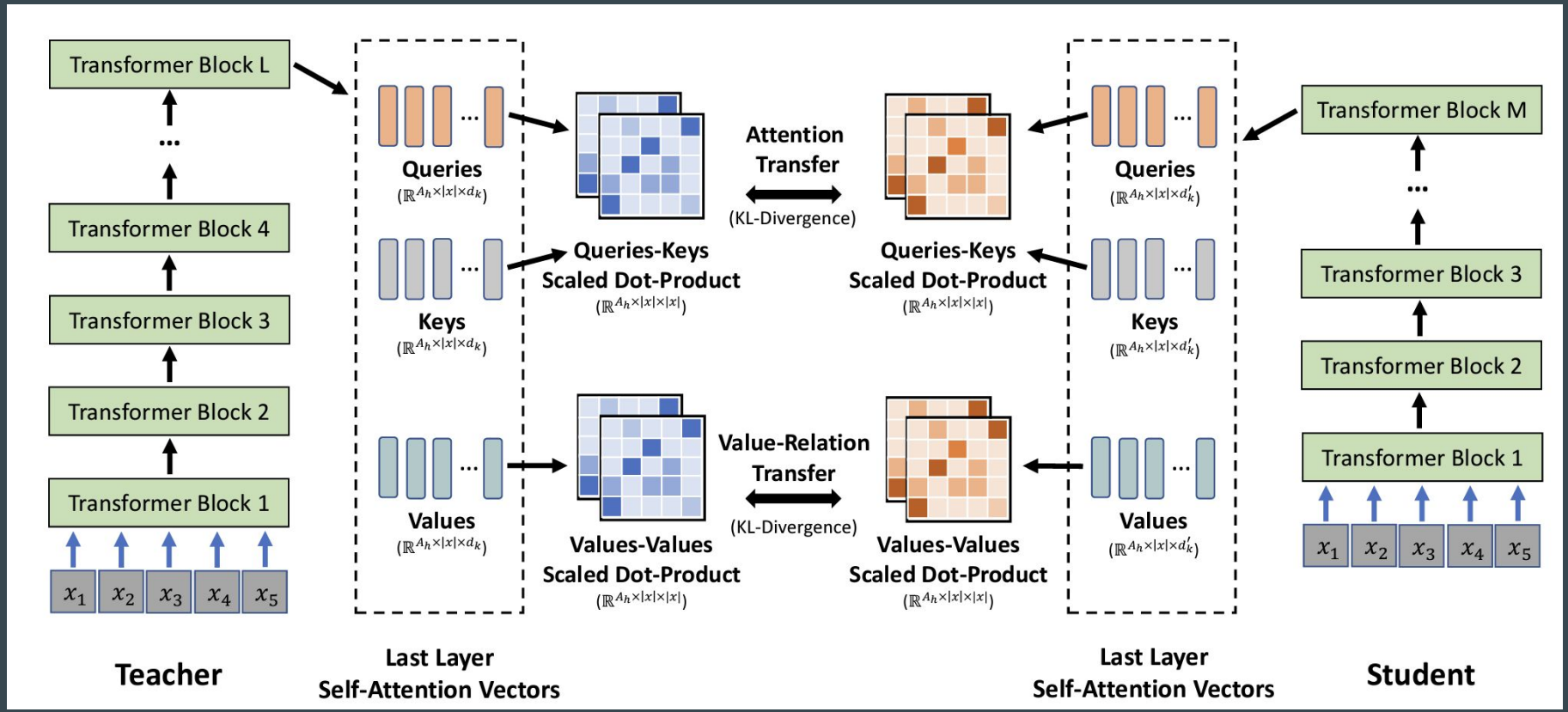
They use KL Divergence between Q-K dot product (refer image next slide) of the teacher and the student model.

Why only the last self attention layer?

- Jawahar et. al - Self attention model of pre-trained LM have rich linguistic information.

- Just taking the last attention layer gives more flexibility to design the student architecture
  - This also avoids the effort of finding the best layer mapping (🤦)

Diagram explaining the idea



MiniLM: Value dot product

Why use “value” in the attention head?

- Introducing relation between the value of student and teacher allows the student to deeply mimic the teacher’s attention behaviour.

Why Scaled Dot product?

- Allows vectors of different hidden dimension into matrix with same size (here- R^A x $|\mathcal{A}|$ x $|\mathcal{A}|$) so now student can use more flexible hidden layer size. And this avoids an additional parameter needed to transform the student representation.

MiniLM: Teacher assistant

In case if student layers < ½ Teacher layers - Prefer to use Teacher assistant to bridge the gap.

Empirically it is seen that using TA helps improve the model performance.

(No comments on what should be the size of TA)

MiniLM: Results

6 layered 768 dimension

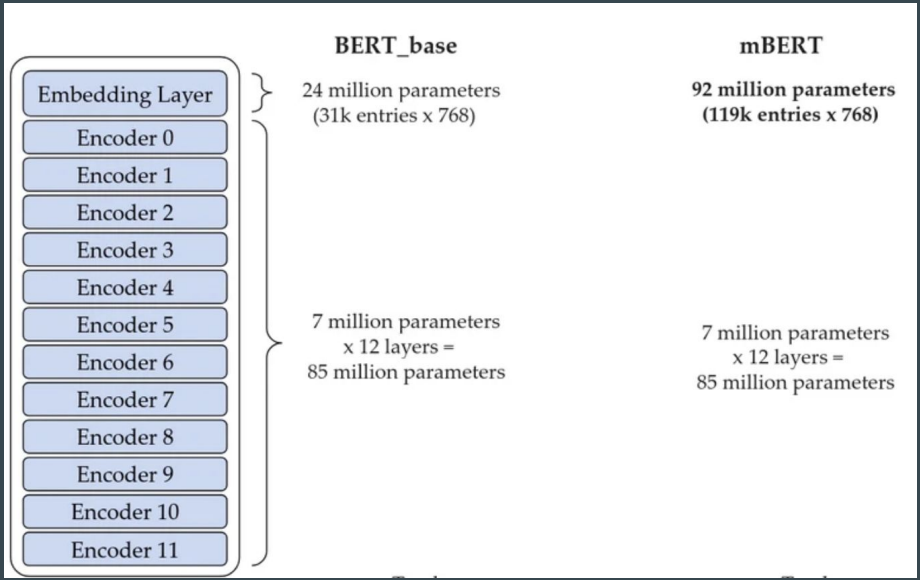
Model	#Param	SQuAD2	MNLI-m	SST-2	QNLI	CoLA	RTE	MRPC	QQP	Average
$\text{BERT}_{\text{BASE}}$	109M	76.8	84.5	93.2	91.7	58.9	68.6	87.3	91.3	81.5
DistillBERT	66M	70.7	79.0	90.7	85.3	43.6	59.9	87.5	84.9	75.2
TinyBERT	66M	73.1	83.5	91.6	90.5	42.8	**72.2**	**88.4**	90.6	79.1
MINILM	66M	**76.4**	**84.0**	**92.0**	**91.0**	**49.2**	71.5	**88.4**	**91.0**	**80.4**

Architecture	#Param	Model	SQuAD 2.0	MNLI-m	SST-2	Average
M=6;d'_h=384	22M	MLM-KD (Soft-Label Distillation)	67.9	79.6	89.8	79.1
		TinyBERT	71.6	81.4	90.2	81.1
		MINILM	72.4	82.2	91.0	81.9
		MINILM (w/ TA)	**72.7**	**82.4**	**91.2**	**82.1**
M=4;d'_h=384	19M	MLM-KD (Soft-Label Distillation)	65.3	77.7	88.8	77.3
		TinyBERT	66.7	79.2	88.5	78.1
		MINILM	69.4	80.3	90.2	80.0
		MINILM (w/ TA)	**69.7**	**80.6**	**90.6**	**80.3**
M=3;d'_h=384	17M	MLM-KD (Soft-Label Distillation)	59.9	75.2	88.0	74.4
		TinyBERT	63.6	77.4	88.4	76.5
		MINILM	66.2	78.8	89.3	78.1
		MINILM (w/ TA)	**66.9**	**79.1**	**89.7**	**78.6**

Load What you need

Why this work is interesting! 🤔

Nearly 52% of the parameters in the MMLMs are stored in embedding layers (in BERT 22%)



And in cases where we just want to use it on a set of languages - Why bother keeping all language parameters in embedding layer ?

Load what you need: Vocab

Steps to make smaller models -

1) **Identify the Vocabulary** - Choose the tokens which appear in at least 0.05% of the paragraphs
2) **Reconstruct the embedding layer** - paper simply mentions: using the new vocab we extract the embeddings for the smaller model.

When we inspect their code -

```
# Copy weights
i = 0
j = 0
vocab = []
for token in old_vocab:
    if token in new_vocab:
        vocab.append(token)
        new_embeddings.weight.data[i, :] = old_embeddings.weight.data[j, :]
        i += 1
    j += 1
```

Language	#Selected tokens	Proportion of original tokens
English (en)	28458	23.8%
French (fr)	24482	20.5%
Spanish (es)	26346	22.0%
German (de)	26031	21.8%
Greek (el)	11616	9.7%
Bulgarian (bg)	12121	10.1%
Russian (ru)	14270	11.9%
Turkish (tr)	19086	16.0%
Arabic (ar)	7292	6.1%
Vietnamese (vi)	17512	14.6%
Thai (th)	8493	7.1%
Chinese (zh)	12928	10.8%
Hindi (hi)	5664	4.7%
Swahili (sw)	16619	13.9%
Urdu (ur)	8656	7.2%
Union (15 langs)	**71577**	**60%**

Note: Union of all language tokens is not the sum (100%) as some tokens are shared among languages

Model	#parameters
bert-base-multilingual-cased (104 languages)	178 million
Geotrend/bert-base-15lang-cased (en, fr, es, de, el, bg, ru, tr, ar, ...)	141 million (-21%)
Geotrend/bert-base-en-fr-cased*	112 million (-37%)

Load what you need: Results

- No performance drop in LWYN
- en-xx : Bilingual models

Models	#params	en	fr	es	de	el	bg	ru	tr	ar	vi	th	zh	hi	sw	ur	avg
Fine-tune multilingual model on English training set (Cross-lingual Transfer)																	
mBERT	178 M	82.1	73.8	73.9	70.3	**66.7**	67.8	68.4	60.4	64.5	70.7	53.0	68.2	**59.5**	**50.3**	57.0	65.8
DistilmBERT	135 M	78.5	70.2	70.1	65.3	60.4	63.1	63.9	55.8	58.6	57.4	37.2	64.2	51.6	46.7	53.3	59.7
$mBERT_{15langs}$	141 M	**82.2**	**74.1**	73.7	70.2	66.3	**68.1**	68.7	**61.1**	64.9	70.6	53.1	**68.8**	58.9	49.0	57.1	65.8
$mBERT_{en-xx}$	108-113 M	**82.2**	73.8	**75.0**	**71.6**	65.3	**68.1**	**69.1**	60.1	**65.0**	**70.9**	**53.2**	68.2	**59.5**	49.1	**57.9**	**65.9**

Limitations

As rest of the model architecture remains the same - inference time ⏱️ is same as the original large model, just the memory (size) of the student model is reduced.

Model	Size (MB)	Memory (MB)	Loading (sec)	Inference (sec)
mBERT	714	1401	4.18	0.24
DistilmBERT	542	1070	3.08	**0.12**
$mBERT_{15langs}$	564	1098	3.14	0.24
$mBERT_{en\_xx}$	445	860	2.76	0.24
$mBERT_{xx}$	**393**	**760**	**2.45**	0.24

Authors have shared the code for reducing any model (having vocab file - so no XLM-R) - Link

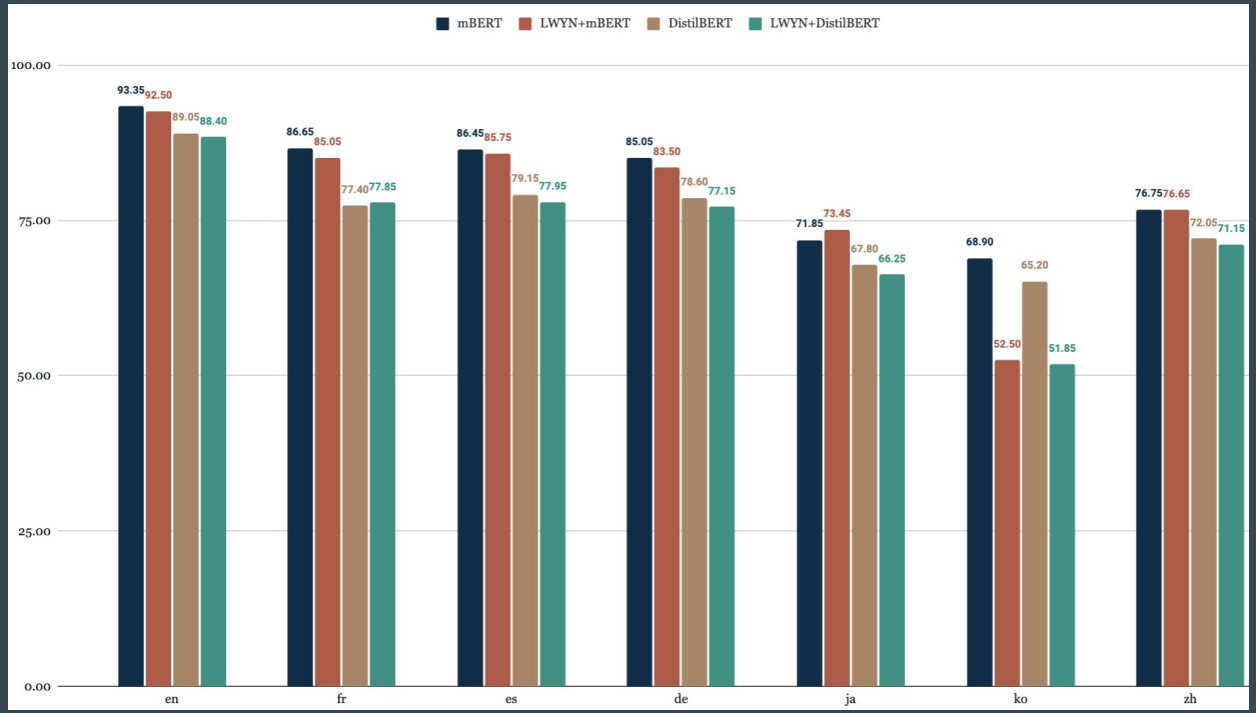
Our Results

What we try to explore?

- DistilBERT + Load what you need : Distilling an already distilled model
- MiniLM vs DistilBERT performance: on trained and zero shot setting
- Performance of mBERT, DistilmBERT, multilingual-MiniLM on tasks like XNLI, Ty-DiQA, PAWS-X
- %age drop of teacher vs student model (mBERT vs DistilmBERT & XLM-R vs multilingual-MiniLM)

Some interesting results

When fine-tuned on English (Data: PAWS-X)

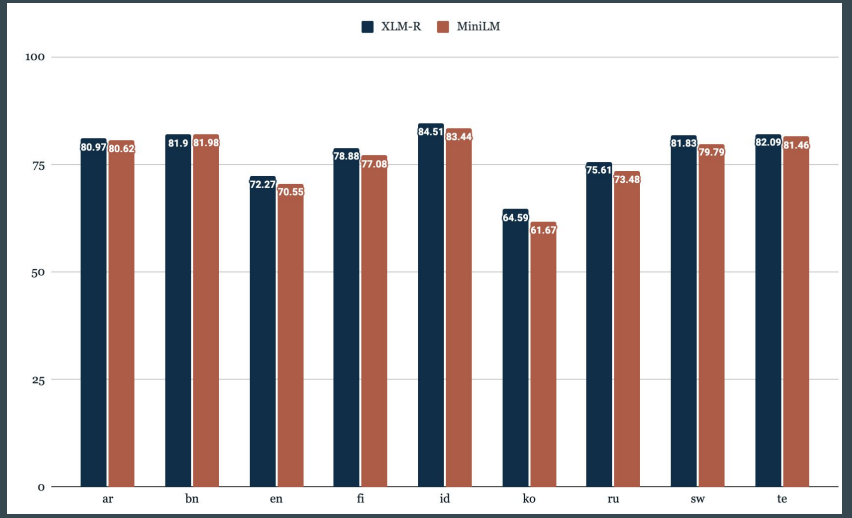


Observation:

- Drop for LWYN is minimal
- Zeroshot performance is decent with similar languages like fr, es when trained on English, with substantial drop in languages like ja, ko

Note: Major drop in zero shot performance when fine-tuned on Korean (refer slides)

Some interesting results



All data fine-tuning (comparison of XLM-R & m-MiniLM on

Student & Teacher	Language combination	Average Drop
DistilmBERT vs mBERT	Finetune on Arabic Test on Arabic	4.80%
	Finetune on Arabic Test on Non-Arabic language	45-50%
	Finetune on All Languages	7-10%
Multilingual MiniLM vs XLM-R	Finetune on Arabic Test on Arabic	2.48%
	Finetune on Arabic Test on Non-Arabic language	13-15%
	Finetune on All Languages	1.50%

Massive drop in zero shot performance of distilbert

Links for all the results and graphs -

Overleaf documentation

Slides presentation -