Computational Linguistics and Intellectual Technologies:
Proceedings of the International Conference "Dialogue 2022"

Moscow, June 15–18, 2022

Knowledge Distillation of Russian Language Models with Reduction of Vocabulary

Alina Kolesnikova[*1] kolesnikova.af@phystech.edu
Yuri Kuratov[*1,2] yurii.kuratov@phystech.edu
Vasily Konovalov[1] vaskoncv@phystech.edu
Mikhail Burtsev[1,2] burtcev.ms@mipt.ru

[1]Neural Networks and Deep Learning Lab, MIPT, Dolgoprodny, Russia
[2]AIRI, Moscow, Russia

Abstract

Today, transformer language models serve as a core component for majority of natural language processing tasks. Industrial application of such models requires minimization of computation time and memory footprint. Knowledge distillation is one of approaches to address this goal. Existing methods in this field are mainly focused on reducing the number of layers or dimension of embeddings/hidden representations. Alternative option is to reduce the number of tokens in vocabulary and therefore the embeddings matrix of the student model. The main problem with vocabulary minimization is mismatch between input sequences and output class distributions of a teacher and a student models. As a result, it is impossible to directly apply KL-based knowledge distillation. We propose two simple yet effective alignment techniques to make knowledge distillation to the students with reduced vocabulary. Evaluation of distilled models on a number of common benchmarks for Russian such as Russian SuperGLUE, SberQuAD, RuSentiment, ParaPhaser, Collection-3 demonstrated that our techniques allow to achieve compression from 17× to 49×, while maintaining quality of 1.7× compressed student with the full-sized vocabulary, but reduced number of Transformer layers only. We make our code and distilled models available.

Keywords: language modeling, transformer, knowledge distillation, compact models, Russian
DOI: 10.28995/2075-7182-2022-21-295-310

Дистилляция знаний для русскоязычных моделей с уменьшением словаря

Алина Колесникова[*1] kolesnikova.af@phystech.edu
Юрий Куратов[*1,2] yurii.kuratov@phystech.edu
Василий Коновалов[1] vaskoncv@phystech.edu
Михаил Бурцев[1,2] burtcev.ms@mipt.ru

[1]Лаборатория нейронных систем и глубокого обучения, МФТИ, Долгопрудный, Россия
[2]Институт искусственного интеллекта AIRI, Москва, Россия

Аннотация

На текущий момент языковые модели типа Трансформер являются основным компонентом для большинства задач обработки естественного языка. Промышленное применение таких моделей требует минимизации времени вычислений и объема памяти. Дистилляция знаний - один из подходов к решению этой задачи. Существующие методы в этой области в основном ориентированы на уменьшение количества слоев или размерности эмбеддингов/скрытых состояний. Другой способ - уменьшить количество токенов в словаре и, следовательно, матрицу эмбеддингов модели-студента. Основной проблемой, которая возникает при уменьшении размера словаря, является несоответствие между входными последовательностями и предсказываемыми распределениями классов моделями учителя и студента. В результате невозможно напрямую применить дистилляцию знаний на основе KL. Мы предлагаем два простых и в тоже время эффективных метода выравнивания, чтобы применить дистилляцию знаний в студента с уменьшенным словарем. Оценка дистиллированных моделей на нескольких распространенных русскоязычных бенчмарках, таких как Russian SuperGLUE, SberQuAD, Rusentiment, ParaPhraser, Collection-3 показала, что предложенные методы позволяют сжать модель от 17 до 49 раз, сохраняя при этом качество модели-студента с полноразмерным словарем и уменьшенным количеством Трансформер-слоев, сжатой в 1.7 раз. Дистиллированные модели и код выложены в открытый доступ.

Ключевые слова: языковое моделирование, трансформер, дистилляция знаний, легковесные модели, русский язык

1 Introduction

Pre-trained Transformer language models have been found to be very successful across a wide range of NLP tasks. Most of the recent state-of-the-art models are based on variations of the original Transformer (Vaswani et al., 2017) and different self-supervised pre-training techniques like masked language modeling (Devlin et al., 2019). Such models became very large, starting from hundreds of millions of parameters (Radford et al., 2018; Devlin et al., 2019; Liu et al., 2019) to hundreds of billions (Brown et al., 2020; Smith et al., 2022; Rae et al., 2021; Lin et al., 2021). Large models require lots of computation, memory, and fast accelerators like TPUs/GPUs. It is challenging to use large models in practical applications where prediction time is critical and available disk/RAM is limited.

General approaches like pruning, quantization, and knowledge distillation (KD) were applied to Transformer language models to make them faster and smaller. Pruning (LeCun et al., 1989) removes some weights of the large models with negligible degradation of predictions. Quantization (Gong et al., 2014) reduces weights precision to float16, int8, int4, or even bits. Knowledge distillation (Buciluǎ et al., 2006; Ba and Caruana, 2014; Hinton et al., 2015) (KD) is used to train smaller student model to mimic behaviour of the larger teacher model.

However, in general, knowledge distillation relies on Kullback-Leibler (KL) divergence over teacher and student predictions. Language models are trained to predict tokens probability distribution in a vocabulary. It implies that teacher and student should share the same vocabulary. If a teacher and student models have different vocabularies, KL loss can not be directly applied as they have different sets of prediction classes. It makes KL-based knowledge distillation for models with mismatched vocabularies impossible. Another outcome of mismatched vocabularies is different tokenization for teacher and student models. It leads to different lengths of input and, therefore, output sequences, which also adds ambiguity to KL-based distillation in this case.

A ratio of embeddings parameters becomes larger as student models become smaller by reducing the number of Transformer layers and/or dimension of hidden representations. Embeddings can get over 50% of all parameters for small models as shown on Figure 1.

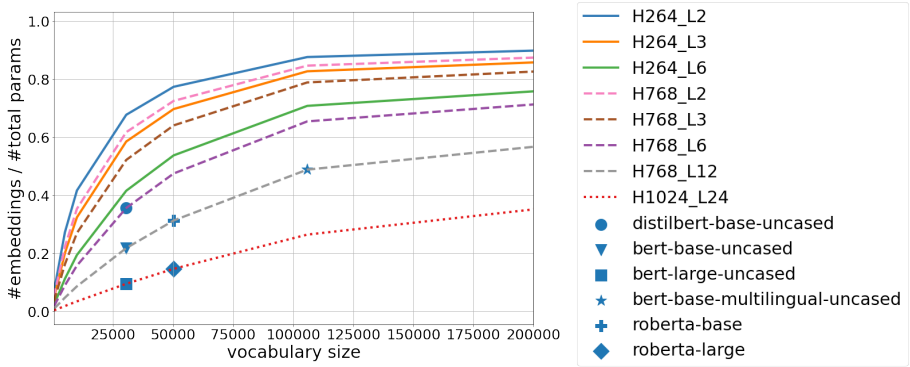


Figure 1: Ratio of number of parameters for embeddings to the full model. In smaller models embeddings have higher fraction of parameters compared to other models with the same vocabulary size. Selected models for English language are shown with the blue markers. The models are denoted by size of hidden representation (H) and number of layers (L).

One of the possible ways to reduce a fraction of embeddings parameters is to make the size of student vocabulary smaller. Moreover, changing student vocabulary could be reasonable for distilling to another domain or from multilingual to monolingual models. Changing student vocabulary leads to the problem of knowledge distillation with mismatched vocabularies.

This paper focuses on applying knowledge distillation to train student models with a smaller vocabulary than the teacher. We propose several strategies for output/intermediate representations alignment. The first one uses teacher and student representations corresponding to the tokens found in both vocabu-

laries (*match* strategy). The second aligns the sequences, produced by student tokenizer, with the teacher by aggregating representations corresponding to an alignment (*reduce* strategy).

We show that teacher's knowledge can be effectively transferred to the student with mismatched vocabulary. We pre-train student models with proposed KD methods and evaluate them on a number of common benchmarks for the Russian language such as Russian SuperGLUE, SberQuAD, RuSentiment, ParaPhraser, and NER on Collection-3. Our students are from $17\times$ to $49\times$ and up to $104\times$ faster on GPU than the teacher while having competitive quality to the $1.7\times$ compressed student. We make our code [1] and pre-trained models[2] available online.

2 Related work

Knowledge distillation can be used to train task-specific fine-tuned and general pre-trained models. Task-specific distillation (Chia et al., 2018; Sun et al., 2019; Tang et al., 2019; Aguilar et al., 2020) takes two steps: fine-tuning a teacher model on a task and distilling it to a student model. The disadvantage of such approach is that it requires repeating both steps for each new task. Large teacher model fine-tuning could be too expensive.

Such models as DistilBERT (Sanh et al., 2019), TinyBERT (Jiao et al., 2020), MobileBERT (Sun et al., 2020b), MiniLM (Wang et al., 2020; Wang et al., 2021) use general pre-training distillation. Distillation could be performed only once to pre-train a general student model, and then student model fine-tunes on tasks, removing expensive teacher fine-tuning step. DistilBERT uses a triple loss: distillation loss between student and teacher output probabilities, student masked language modeling loss, and cosine loss for hidden representation of student and teacher models. TinyBERT adds trainable student-teacher projections for embeddings and Transformer layer output representations. These projections allow training student models with Transformer layer hiddens of arbitrary size. TinyBERT, MobileBERT, MiniLM use attention matrices as an additional source of knowledge for distillation. A student model trains to produce similar attention matrices to a teacher by additional loss term.

However, previously mentioned pre-training knowledge distillation approaches are not flexible enough. Student model vocabulary should be the same as a teacher model to compute the distillation loss. Different vocabularies also lead to different tokenization, hence different student and teacher sequence lengths. Therefore, student-teacher output probabilites, hiddens and attention matrices are not aligned to be used with losses mentioned above.

(Zhao et al., 2021) addresses these problems with mixed-vocabulary training. Authors propose first to pre-train student embedding matrix together with teacher model and then use it for regular student model MLM pre-training. Tokenization for each word in mixed-vocabulary training is performed by randomly selecting teacher or student vocabulary with corresponding embeddings matrix. This way, only the embeddings matrix is trained using teacher model knowledge. All other parameters of the smaller student model do not benefit from the teacher model. In our work, we propose methods that allow training student model with knowledge distillation in one stage and using teacher knowledge from all layers.

The problem with mismatched vocabulary is actually more general. Vocabulary tokens are essentially labels for a token classification task, that is, language modeling. In other words, the more general problem is knowledge distillation for teacher and student models with different sets of labels. Instead of predictions distillation, pre-classification layer outputs or other representations might be used for distillation (Tian et al., 2020; Sun et al., 2020a), making a connection to representation-based learning (Bromley et al., 1993; Chen et al., 2020).

Alternatively, the number of parameters in Transformer models can be reduced by parameters sharing (Lan et al., 2020), embeddings matrix factorization (Sun et al., 2020b; Hrinchuk et al., 2019; Lan et al., 2020), and pruning (Voita et al., 2019; Gordon et al., 2020). These approaches are complementary to knowledge distillation in general and to our methods as well.

[1] github.com/ayeffkay/rubert-tiny

[2] See models with `distil-` prefix at huggingface.co/DeepPavlov

3 Distillation strategy

3.1 Background

One of the first attempts of pre-trained Transformer language model distillation is DistilBERT (Sanh et al., 2019). Authors introduce training objective which is a linear combination of the supervised masked language modeling (MLM) loss:

$$\mathcal{L}_{mlm} = - \sum_{\substack{i=1, \\ i \in \texttt{masked\_ids}}}^{|X_t|} \sum_{j=1}^{|V_t|} y_{ij} \log p_{ij}^s, \tag{1}$$

distillation loss over the soft target probabilities of the teacher:

$$\mathcal{L}_{ce} = - \sum_{\substack{i=1, \\ i \notin \texttt{masked\_ids}}}^{|X_t|} \sum_{j=1}^{|V_t|} p_{ij}^t \log p_{ij}^s, \tag{2}$$

and cosine distance loss for the student and teacher hidden representations:

$$\mathcal{L}_{cos} = \sum_{\substack{i=1, \\ i \notin \texttt{masked\_ids}}}^{|X_t|} \texttt{cos\_dist}\left(h_{n,i}^t, h_{m,i}^s\right), \texttt{cos\_dist}\left(h_{n,i}^t, h_{m,i}^s\right) = 1 - \frac{\langle h_{n,i}^t, h_{m,i}^s \rangle}{\|h_{n,i}^t\| \|h_{m,i}^s\|}, \tag{3}$$

here `masked_ids` is a set of subword indices, masked with some probability; $|X_t|$ is a subwords sequence length obtained after input sequence tokenization by teacher tokenizer; V_t is a teacher vocabulary with the size $|V_t|$; y_{ij} is a masked subword index in a vocabulary; p^s, p^t are subword probabilities produced by student and teacher models; h_m^s, h_n^t are student and teacher hidden states taken from m-th and n-th Transformer layers. Alternatively Kullback-Leibler divergence can be used instead of $\mathcal{L}_{ce}$.[3]

Usually, it is assumed that a teacher and a student use the same vocabulary, i.e. inputs for the teacher and the student will match after tokenization. But if a teacher and a student use different vocabularies, then tokenized inputs will be different and will not always have the same length. Further we represent the problem statement more formally and provide our solutions.

3.2 Problem statement

Given teacher with vocabulary V_t and student with vocabulary V_s, such that $|V_s| < |V_t|$, $V_s \cap V_t \neq \varnothing$ [4]. Then LM output probabilities shapes will be $(|X_t|, |V_t|)$, $(|X_s|, |V_s|)$ and hidden states shapes will be $(|X_t|, d_t)$, $(|X_s|, d_s)$ for teacher and student respectively, where X_t and X_s are inputs produced after tokenization by teacher and student tokenizers, d_s and d_t are hidden states dimension. As mentioned above, in general $X_t \neq X_s$ and $V_t \neq V_s$. The task is *to define alignment* $\mathcal{X} : X_s \to X_t$ *for sequence length dimension* to obtain $|X_s| = |X_t|$ and *mapping* $\mathcal{V} : V_s \to V_t$ *between vocabularies*.

For simplicity we will assume that $|X_t| \leq |X_s|$. The rationale behind this lies in the observation that because of reduced vocabulary size $|V_s|$ BPE tokenization algorithm will keep less amount of more frequent subwords, thus leading to longer student-generated outputs. Our observation confirmed when we compared sequence lengths produced by teacher and student pre-trained tokenizers with $\sim 1.2 \times 10^5$ and $\sim 3 \times 10^4$ vocabulary sizes respectively. On the corpus of $\sim 2.7 \times 10^7$ sequences only 0.2% of the student-tokenized sequences were shorter than the teacher.

[3] The difference will be in the term $p_{ij}^t \log\left(p_{ij}^t\right)$.

[4] We emphasize that *non-empty intersection condition between teacher and student vocabularies is necessary*, because the strategies below cannot be applied in the case of its complete mismatch.

3.3 Sequence length dimension alignment

We propose two strategies for sequence length dimension alignment: *match* strategy and *reduce* strategy. The first one can be applied *both* to the sequence length and vocabulary dimension, the second one for sequence length dimension only.

For the *match* strategy, after building student vocabularies of sizes 5×10^3, 10^4, 2×10^4, 3×10^4 using BPE subword tokenization algorithm we found that $\sim 99\%$ student subwords are in the teacher vocabulary of $\sim 1.2 \times 10^5$ size. Therefore, we can take into account only matching subwords and mask all mismatched subwords (Figure 2)

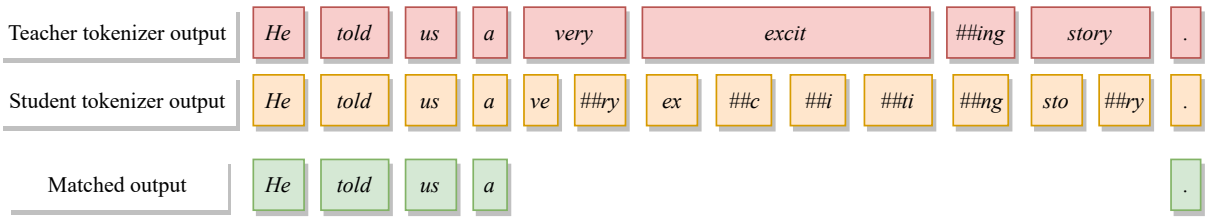


Figure 2: Match strategy for sequence length dimension alignment. The first sequence is produced by the teacher's tokenizer, and the second by the student's.

If n_{match} subwords in sequence and $|V_{match}|$ subwords in vocabulary match, then hidden states and output LM probabilities shapes are transformed as follows:

$$\begin{aligned} (|X_t|, d_t) &\mapsto (n_{match}, d_t)\,, \quad (|X_t|, |V_t|) \mapsto (n_{match}, |V_{match}|)\,, \\ (|X_s|, d_s) &\mapsto (n_{match}, d_s)\,, \quad (|X_s|, |V_s|) \mapsto (n_{match}, |V_{match}|)\,. \end{aligned} \tag{4}$$

This makes sequences equal by length and aligned for teacher and student models. LM output probabilities also have equal shapes. KL or CE losses can be used for distillation now with *match* strategy.

It can be seen that *match* strategy lowers overhead required to compute losses. The main drawback is that we lose from 75% (for 3×10^4 vocabulary size) to 96% (for 5×10^3 vocabulary size) subwords that can be used for distillation from the teacher. Another drawback is that embeddings corresponding to the matching subwords might occur in different contexts for teacher and student and thus might cover different meanings.

In general the task of finding correspondence between teacher- and student-tokenized sequences is ambiguous. For example in Figure 2 depending on the tokenizer we can obtain highly mismatched subword sequences:

(5) (Teacher) *excit ##ing*
(Student) *ex ##c ##i ##ti ##ng*

In *reduce* alignment strategy an auxiliary input for a student model receives teacher subwords greedily split into student subwords from left to right. Then student's intermediate/output representations corresponding to the one teacher subword are aggregated by summation as shown on Figure 3. Assume that i-th teacher subword was splitted by student subwords with indices $k_1^i, k_2^i, \ldots, k_l^i$. Then formally, aggregation procedure for hidden states can be written as follows:

$$h_i^t = \sum_{j \in \{k_1^i, \ldots, k_l^i\}} h_j^s,\ i = \overline{1, |X_t|} \tag{6}$$

Pre-softmax outputs aggregation procedure can be represented in a similar way.

This allows the student to learn mapping from the teacher's vocabulary.

Reduce strategy leaves teacher representations shapes unchanged, and for the student we obtain sequence aligned to teacher sequence length:

$$(|X_s|, d_s) \mapsto (|X_t|, d_s)\,, \quad (|X_s|, |V_s|) \mapsto (|X_t|, |V_s|) \tag{7}$$

This can be combined with the match strategy, if vocabulary alignment needed:

$$(|X_t|, |V_s|) \mapsto (|X_t|, |V_{match}|)\,. \tag{8}$$

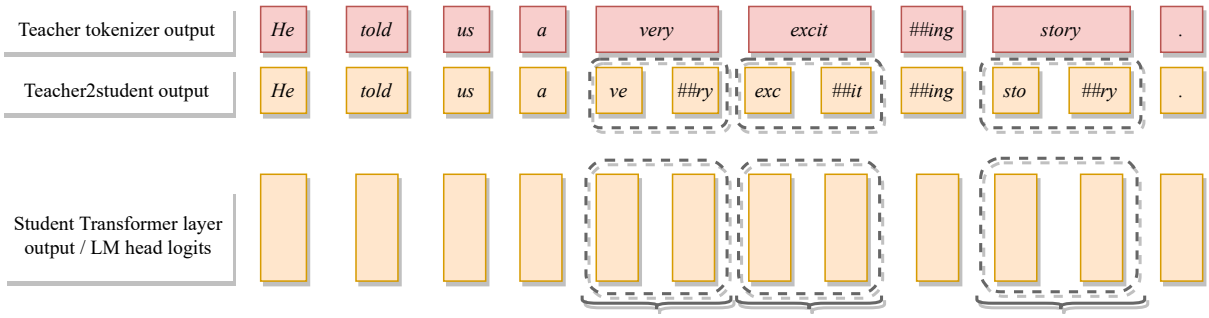


Figure 3: Reduce strategy. The first sequence is an output from the teacher's tokenizer, the second is a greedy split result of the first sequence by subwords from the student's vocabulary.

Reduce strategy combined with *match* over vocabulary introduces another way to use KL or CE losses for distillation. Compared to the match strategy only, we can use all teacher outputs and representations, so reducing the student sequence allows to extract knowledge for all tokens of the teacher vocabulary. But this approach still skips subwords from the student's vocabulary which are not found in the teacher's. This can be partially offset by passing two inputs to the student model. The first one is a teacher-to-student split with subsequent reduction to compute distillation losses, and the second output from the student's tokenizer to compute supervised masked language modeling loss. The drawbacks of reduce compared to match strategy are higher overhead to compute losses and greedy split which might be not optimal.

4 Experiments

4.1 Pre-training

Corpus Teacher pre-training and distillation to the students was made on the same Russian Language data of $\sim$ 27M sentences collected from OpenSubtitles (Lison and Tiedemann, 2016), Dirty & Pikabu web resourses, and Social Media segment of Taiga corpus (Shavrina and Shapovalova, 2017).

Models We used pre-trained rubert-base-cased-conversational (12-layer Russian BERT model)[5] as a teacher with hidden states dimension of 768 and vocabulary size of 120K. It was the largest and the slowest model in our experiments.

Two students `distil-base` [6] and `distil-small`[7] have the same vocabulary and dimension of hidden states as the teacher, but a number of Transformer layers were reduced to 6 and 2. To train distil-base and `distil-small` we extended the distillation strategy proposed for DistilBERT (Sanh et al., 2019). Namely, we added MSE loss for averaged attention maps and cosine distance loss for averaged hidden states. To average teacher attention maps and hidden states, we grouped them by six Transformer layers for 2-layer `distil-tiny` and by two for 6-layer `distil-base` (because the teacher model has 12 Transformer layers).

Models `distil-tiny(30|20|10|5)` were students with 3 Transformer layers, hidden states dimension of 264 and reduced vocabulary sizes of 30k, 20k, 10k and 5k. We applied proposed alignment strategies to `distil-tiny*` models.

We compare proposed distilled models to other available state-of-the-art distilled models for Russian `rubert-tiny` and `rubert-tiny2`[8]. Models `rubert-tiny` and `rubert-tiny2` are 3-layer Trans-

[5] huggingface.co/DeepPavlov/rubert-base-cased-conversational
[6] huggingface.co/DeepPavlov/distilrubert-base-cased-conversational
[7] huggingface.co/DeepPavlov/distilrubert-tiny-cased-conversational
[8] huggingface.co/cointegrated/rubert-tiny, huggingface.co/cointegrated/rubert-tiny2, habr.com/ru/post/562064/

formers distilled from multiple teachers and combining MLM and Translation Ranking Modeling (TLM, (Feng et al., 2020)) losses.

All models that we trained and evaluated are listed in Appendix A Table 3 with corresponding inference speed and memory requirements.

Distillation with reduced vocabulary We distilled the teacher model into 3-layer student model `distil-tiny30` with 30k subwords in vocabulary. We tried different combinations of loss functions and alignment strategies. Combinations are summarized in Figure 4. In our experiments, we use MLM loss in summation with KL or MSE, or both of them. To compute KL loss teacher and student pre-softmax outputs should be aligned: 1. with the match strategy by sequence and vocabulary (`KL-match`); 2. with the reduce-match strategy, where reduction was made by sequence dimension and match-by vocabulary (`KL-reduce-match`). MSE loss for hidden states distillation was applied with match and reduce strategies. To match hidden sizes projection layers were used (see details in Appendix B.2).

To apply reduce strategy, we passed two inputs to the student: 1. a student-tokenized input for MLM loss; 2. teacher inputs tokenized by student for MSE and KL. Student representations corresponding to the student-tokenized input *were not aligned and were not used* to compute MSE or KL divergence.

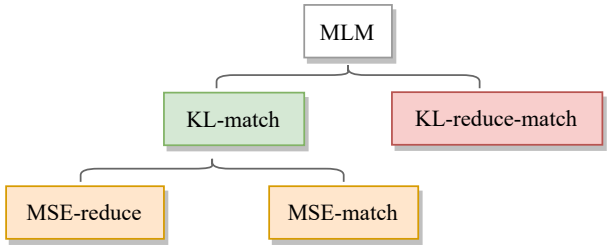


Figure 4: Combinations of loss functions (MLM, KL, MSE) and alignment strategies (reduce, match, reduce-match).

We initialized student models' embeddings by re-using teacher embeddings (see details in Appendix B.1). Other training details could be found in Appendix B.

Ablation To check whether match and reduce strategies are effective for distilling knowledge from the teacher, we pre-trained `distil-tiny30` using only MLM loss and without any distillation losses. We also performed pre-training without KL divergence loss term to evaluate its contribution to `KL-match & MLM & MSE` combination.

To evaluate effect of reduced vocabulary on the distillation quality, we compared `distil-tiny*` models with reduced vocabulary to `distil-base` and `distil-small` with the same vocabulary as the teacher.

To determine how further vocabulary size reduction affects the distillation quality, we also distilled teacher into `distil-tiny` models with 5k, 10k and 20k vocabulary sizes (results are in Appendix C).

4.2 Fine-tuning

For evaluation we fine-tuned our models on ParaPhraser (Pivovarova et al., 2017), RuSentiment (Rogers et al., 2018), SberQuAD (Efimov et al., 2020), NER Collection-3 (Mozharova and Loukachevitch, 2016) and Russian SuperGLUE (Shavrina et al., 2020) datasets. Their description is given in Appendix D and Table 5.

Results for ParaPhraser, Collection-3, RuSentiment and SberQuAD are collected in the Table 1. From Table 1 we see expected result that pre-training with MLM is better than random initialization for further fine-tuning. Also, pre-training with distillation improves student models.

Results for the best distilled models on RussianSuperGLUE test sets are shown in the Table 2. We use the following naming conventions for `distil-tiny30` models for RussianSuperGLUE results:

- `MLM & KL & MSE (RT)` with reduce strategy and trainable hiddens projections is `distil-tiny-1`;
- `MLM & KL & MSE (MT)` with the same losses combination and match strategy is `distil-tiny-2`;

• `MLM & MSE (RF)` with reduce strategy and frozen projections for hidden states is `distil-tiny-3`.

We selected `MLM & KL & MSE (MT)` over `MLM & KL & MSE (RF)` despite the better average performance as this difference is caused by SberQuAD scores only. On the other datasets `MLM & KL & MSE (MT)` performs better or almost the same as `MLM & KL & MSE (RF)`.

Logits distillation with KL divergence loss Proposed *match* and *reduce-match* strategies to align pre-softmax outputs of the teacher and student models improve results obtained by MLM pre-training only. Results from Table 1 show that *match* strategy performs better than *reduce-match*. Reduction of logits via summing might not result in the true probability of subword compounding of reduced subwords. The teacher model pre-training procedure does not guarantee that subword probability would be equal to multiplication of its compounding subwords probabilities.

Hidden states distillation Distilling hidden states with MSE loss can improve `KL-match & MLM` combination. On average, reduce strategy for hidden states alignment works better than match in combinations with KL divergence and without it. Distilling from hidden states allows extracting more knowledge from the teacher and its intermediate states. This observation holds for both the results in Table 1 and Russian SuperGLUE in Table 2.

Surprisingly, frozen projections, that is, non-trainable random projections, perform better for some of the configurations than trainable. For SberQuAD dataset, frozen projections steadily show higher F1 and EM scores, e.g., improving F1 for trainable projections from $+1$ to $+26$ F1 points. Though the result is not expected, it has also been previously observed that random projections could be very effective (Wieting and Kiela, 2019).

Model	Proj	Distillation Losses	ParaPhraser	RuSentiment	Collection-3	SberQuAD	
			F1	F1 (weighted)	Entity F1	F1	EM
teacher	-	MLM, NSP	**86.30**±0.96	**76.00**±0.53	**97.01**± 0.13	**83.82**±0.15	**65.60**±0.12
distil-base	-	MLM, KL, MSE, Cos	82.86±0.47	75.82±0.98	96.40±0.20	80.05±0.43	60.96±0.51
distil-small	-		75.53±1.03	74.58±0.10	94.20±0.20	68.92±0.30	48.21±0.39
distil-tiny30	-	-	72.48±0.32	69.27±0.35	75.61±0.41	17.54±0.09	4.46±0.14
	-	MLM	**74.54**±0.20	**71.68**±0.30	**92.04**±0.26	**38.17**±0.21	**22.12**±0.30
	M	MLM, KL	**74.59**±0.20	72.90±0.20	**93.19**±0.17	**52.64**±0.37	**34.74**±0.41
	RM		74.40±0.23	**72.98**±0.19	93.01±0.11	38.41±0.54	22.20±0.51
	MF	MLM, KL, MSE	**75.27**±0.20	73.06±0.21	93.30±0.14	49.43±1.83	31.33±1.69
	MT		74.99±0.20	73.38±0.20	93.52±0.11	53.14±0.35	35.85±0.47
	RF		74.68±0.20	73.27±0.20	93.28±0.09	**60.26**±0.55	**40.82**±0.61
	RT		75.06±0.20	**73.70**±0.20	**93.71**±0.10	55.02±0.62	36.28±0.62
	MF	MLM, MSE	74.56±0.20	72.80±0.20	92.64±0.13	42.62±0.62	25.85±0.51
	MT		74.25±0.30	73.11±0.23	93.06±0.11	43.37±0.38	26.08±0.49
	RF		**75.23**±0.17	**73.45**±0.17	**93.87**±0.09	**69.03**±0.24	**48.46**±0.36
	RT		74.81±0.16	73.12±0.27	93.26±0.12	43.26±0.73	26.29±0.54
rubert-tiny	T	MLM, TLM, MSE	74.36±0.23	69.34±0.22	91.23±0.17	39.74±0.52	23.70±0.48
rubert-tiny2			**78.72**±0.15	**71.84**±0.24	**93.72**±0.11	**67.80**±0.22	**47.64**±0.32

Table 1: Fine-tuning results for ParaPhraser, RuSentiment, Colleciton-3 and SberQuAD. "Proj" column means type of alignment (first letter, match-M, reduce-R) and projection mode for hidden states (second letter, frozen-F, trainable-T). RM means reduce-match combination for KL loss. Empty "Losses" cell is to denote student without pre-training.

Distillation without KL divergence loss Surprisingly, the best of `distil-tiny30` students are `MLM & MSE (RF)` with reduce strategy and frozen projections did not use KL loss at all. `MLM & MSE (RF)` is very close by quality to `distil-small` and `rubert-tiny2`, requiring $10\times$ (resp. $3\times$) less memory and being $2\times$ (resp. $5\times$) faster on CPU. Moreover, for datasets from Table 1, except SberQuAD, losses combinations without KL divergence work very close to combinations with it. This result also holds on majority of Russian SuperGLUE tasks.

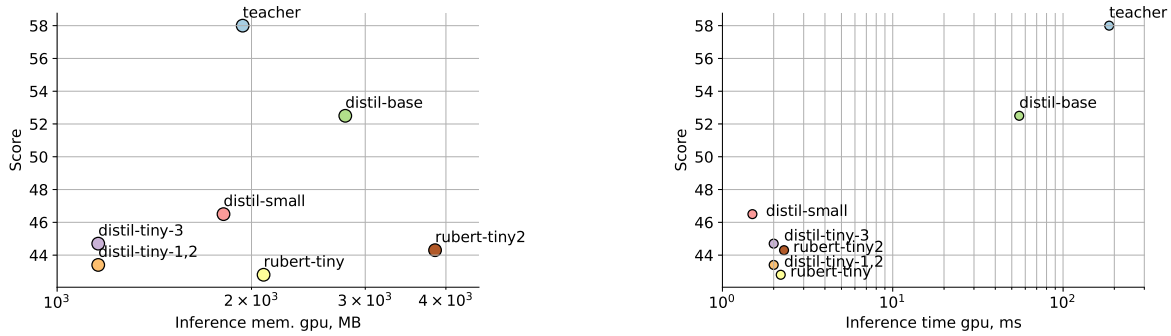
The low impact and inefficiency of KL-loss for distillation might be due to *match* a shift in matching subwords meanings in student and teacher vocabulary. But we do not have a solid proof for that and further investigation is needed.

Model	Score	LiDi	RCB	PARus	MuSeRC	TERRa	RUSSE	RWSD	DNQA	RuCoS
		Mcorr.	F1/Acc.	Acc.	F1a/EM	Acc.	Acc.	Acc.	Acc.	F1/EM
teacher	**54.8**	**21.2**	31.1/**50.8**	57.2	**67.5/27.1**	**51.4**	**71.1**	62.3	63	**79/78.5**
distil-base	49.84	8.5	33.0/47.1	61.0	51.1/6.3	49.5	63.5	63.0	**65.5**	69.0/68.6
distil-small	45.24	3.7	**34.7**/46.5	**65.8**	48.6/7.8	48.5	55.0	**66.9**	58.6	40.0/39.7
distil-tiny-1	42.63	4.2	28.8/48.9	49.0	40.4/6.6	**53.7**	55.1	63.6	60.4	35.5/35.2
distil-tiny-2	42.86	3.1	25.8/45.4	**53.6**	40.4/6.6	52.4	55.7	63.6	**61.7**	**36.5/36.5**
distil-tiny-3	**44**	**4.6**	**35.0/50.1**	52.7	**43.3/7.4**	52.8	**56.5**	**66.9**	**61.7**	33.0/32.7
rubert-tiny	42	-0.9	31.5/43.0	52.8	**46.5**/9.3	49.6	54.3	**66.5**	**63.8**	27.0/26.7
rubert-tiny2	**45.19**	**17**	**36.7/43.7**	**57.1**	44.5/**9.8**	**50.4**	**59.5**	65.9	58.7	**31.0/30.5**

Table 2: The model performance on the Russian SuperGLUE test sets. Matthews correlation for the LiDiRus task is scaled to $[-100, 100]$.

The results on Russian SuperGLUE partially meet the results on ParaPhraser, Collection-3, RuSentiment and SberQuAD. The teacher model significantly outperforms the rest. However, on PARus and RWSD `distil-small` achieves better results. This might be due to the limited size of the training data. All `distil-tiny*` models achieve comparable results with the `distil-tiny-3`[9] slightly ahead, so the contribution of the KL loss to the student performance is not clear. The `rubert-tiny2` model outperforms `rubert-tiny` confirming the previous results.

Dependence of model score on inference time and memory The dependence of models Russian SuperGLUE score on GPU [10] inference time and memory is shown on Figure 5.



(a) The dependence of score on GPU inference memory

(b) The dependence of score on GPU inference time

Figure 5: The dependence of models Russian SuperGLUE average score on GPU inference time and memory. Random batches of size 16 and sequence length 512 were used.

From Figure 5a we can conclude that memory required for inference and model quality are not always correlated (e.g. `distil-base` and `rubert-tiny2` are worse by quality than the teacher but require more memory). It is caused by differences in the particular implementations of the Transformer architectures. Our `distil-tiny` models have the lowest memory consumption. At the same time from Figure 5b we can conclude that model score and inference time are highly correlated.

[9] We make it available at `https://huggingface.co/DeepPavlov/distilrubert-tiny-cased-conversational-v1`

[10] See hardware details in Appendix A

5 Conclusions and future work

We introduced two language model distillation strategies allowing to reduce student's vocabulary. *Match* strategy uses only representations for the subwords which are common for a teacher and a student vocabularies. *Reduce* strategy aggregates a student's subwords representations corresponding to particular teacher's subwords. We performed experiments to show how vocabulary reduction affects the distillation process and how our strategies can be effectively applied for distillation based on teacher output and intermediate representations. We trained student models of different sizes which are from $1.3\times$ to $49\times$ smaller than the teacher while maintaining a good quality compared to the other SOTA models for Russian of similar size. We found that distillation without Kullback-Leibler divergence loss for models with reduced vocabularies performs the best. Our experiments showed that $17\times$ compressed student with reduced vocabulary can work very close to $1.3\times$ compressed student with the same vocabulary as the teacher. Additionally, we made the best of our models and code to train them publicly available.

As further improvements, we consider other ways of distilling intermediate representations based on contrastive and metric learning approaches as well as the more accurate mapping between mismatched subwords in vocabularies to transfer as much knowledge as possible during the distillation process.

Acknowledgments

This work was supported by a grant for research centers in the field of artificial intelligence, provided by the Analytical Center for the Government of the Russian Federation in accordance with the subsidy agreement (agreement identifier 000000D730321P5Q0002) and the agreement with the Moscow Institute of Physics and Technology dated November 1, 2021 No. 70-2021-00138

Authors' contributions

A.K. suggested some of the experiments, developed and optimized code for training and fine-tuning (except Russian SuperGLUE), carried out the experiments, performed most of the computations, and wrote the manuscript (except Sec. 1, 2). Y.K. suggested the original idea of experiments, helped with code optimization, helped with performing and designing experiments, participated actively in results discussion, wrote Sec. 1, 2 and edited the rest of the manuscript. V.K. performed fine-tuning on the Russian SuperGLUE benchmark, described the results, and suggested edits for the manuscript. M.B. supervised the team, discussed intermediate results and directions of the study, contributed to the manuscript's final version. All authors discussed results and approved the final version of the paper.

References

Gustavo Aguilar, Yuan Ling, Yu Zhang, Benjamin Yao, Xing Fan, and Chenlei Guo. 2020. Knowledge distillation from internal representations. // *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, P 7350–7357.

Jimmy Ba and Rich Caruana. 2014. Do deep nets really need to be deep? // Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Q. Weinberger, *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc.

Jane Bromley, Isabelle Guyon, Yann LeCun, Eduard Säckinger, and Roopak Shah. 1993. Signature verification using a" siamese" time delay neural network. *Advances in neural information processing systems*, 6.

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. // H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, *Advances in Neural Information Processing Systems*, volume 33, P 1877–1901. Curran Associates, Inc.

Cristian Buciluǎ, Rich Caruana, and Alexandru Niculescu-Mizil. 2006. Model compression. // *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, P 535–541.

Mikhail Burtsev, Alexander Seliverstov, Rafael Airapetyan, Mikhail Arkhipov, Dilyara Baymurzina, Nickolay Bushkov, Olga Gureenkova, Taras Khakhulin, Yuri Kuratov, Denis Kuznetsov, Alexey Litinsky, Varvara Logacheva, Alexey Lymar, Valentin Malykh, Maxim Petrov, Vadim Polulyakh, Leonid Pugachev, Alexey Sorokin, Maria Vikhreva, and Marat Zaynutdinov. 2018. DeepPavlov: Open-source library for dialogue systems. // *Proceedings of ACL 2018, System Demonstrations*, P 122–127, Melbourne, Australia, July. Association for Computational Linguistics.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. // Hal Daumé III and Aarti Singh, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, P 1597–1607. PMLR, 13–18 Jul.

Yew Ken Chia, Sam Witteveen, and Martin Andrews. 2018. Transformer to CNN: Label-scarce distillation for efficient text classification. // *NIPS Workshop CDNNRIA 2018*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. // *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)*, P 4171–4186.

Pavel Efimov, Andrey Chertok, Leonid Boytsov, and Pavel Braslavski. 2020. Sberquad–russian reading comprehension dataset: Description and analysis. // *International Conference of the Cross-Language Evaluation Forum for European Languages*, P 3–15. Springer.

Fangxiaoyu Feng, Yinfei Yang, Daniel Cer, Naveen Arivazhagan, and Wei Wang. 2020. Language-agnostic bert sentence embedding. *arXiv preprint arXiv:2007.01852*.

Yunchao Gong, Liu Liu, Ming Yang, and Lubomir Bourdev. 2014. Compressing deep convolutional networks using vector quantization. *arXiv preprint arXiv:1412.6115*.

Mitchell Gordon, Kevin Duh, and Nicholas Andrews. 2020. Compressing BERT: Studying the effects of weight pruning on transfer learning. // *Proceedings of the 5th Workshop on Representation Learning for NLP*, P 143–155, Online, July. Association for Computational Linguistics.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. // *Proceedings of the IEEE international conference on computer vision*, P 1026–1034.

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

Oleksii Hrinchuk, Valentin Khrulkov, Leyla Mirvakhabova, Elena Orlova, and Ivan Oseledets. 2019. Tensorized embedding layers for efficient model compression. *arXiv preprint arXiv:1901.10787*.

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. // *Findings of the Association for Computational Linguistics: EMNLP 2020*, P 4163–4174, Online, November. 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. // *International Conference on Learning Representations*.

Yann LeCun, John Denker, and Sara Solla. 1989. Optimal brain damage. *Advances in neural information processing systems*, 2.

Junyang Lin, Rui Men, An Yang, Chang Zhou, Ming Ding, Yichang Zhang, Peng Wang, Ang Wang, Le Jiang, Xianyan Jia, et al. 2021. M6: A chinese multimodal pretrainer. *arXiv preprint arXiv:2103.00823*.

Pierre Lison and Jörg Tiedemann. 2016. Opensubtitles2016: Extracting large parallel corpora from movie and tv subtitles.

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*.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Valerie Mozharova and Natalia Loukachevitch. 2016. Two-stage approach in russian named entity recognition. // *2016 International FRUCT Conference on Intelligence, Social Media and Web (ISMW FRUCT)*, P 1–6. IEEE.

Lidia Pivovarova, Ekaterina Pronoza, Elena Yagunova, and Anton Pronoza. 2017. Paraphraser: Russian paraphrase corpus and shared task. // *Conference on artificial intelligence and natural language*, P 211–225. Springer.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding with unsupervised learning. Technical report.

Jack W Rae, Sebastian Borgeaud, Trevor Cai, Katie Millican, Jordan Hoffmann, Francis Song, John Aslanides, Sarah Henderson, Roman Ring, Susannah Young, et al. 2021. Scaling language models: Methods, analysis & insights from training gopher. *arXiv preprint arXiv:2112.11446*.

Anna Rogers, Alexey Romanov, Anna Rumshisky, Svitlana Volkova, Mikhail Gronas, and Alex Gribov. 2018. Rusentiment: An enriched sentiment analysis dataset for social media in russian. // *Proceedings of the 27th international conference on computational linguistics*, P 755–763.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Tatiana Shavrina and Olga Shapovalova. 2017. To the methodology of corpus construction for machine learning:«taiga» syntax tree corpus and parser. *Proceedings of the "Corpora*, P 78–84.

Tatiana Shavrina, Alena Fenogenova, Emelyanov Anton, Denis Shevelev, Ekaterina Artemova, Valentin Malykh, Vladislav Mikhailov, Maria Tikhonova, Andrey Chertok, and Andrey Evlampiev. 2020. RussianSuperGLUE: A Russian language understanding evaluation benchmark. // *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, P 4717–4726, Online, November. Association for Computational Linguistics.

Shaden Smith, Mostofa Patwary, Brandon Norick, Patrick LeGresley, Samyam Rajbhandari, Jared Casper, Zhun Liu, Shrimai Prabhumoye, George Zerveas, Vijay Korthikanti, et al. 2022. Using deepspeed and megatron to train megatron-turing nlg 530b, a large-scale generative language model. *arXiv preprint arXiv:2201.11990*.

Siqi Sun, Yu Cheng, Zhe Gan, and Jingjing Liu. 2019. Patient knowledge distillation for bert model compression. // *EMNLP/IJCNLP (1)*, P 4322–4331.

Siqi Sun, Zhe Gan, Yuwei Fang, Yu Cheng, Shuohang Wang, and Jingjing Liu. 2020a. Contrastive distillation on intermediate representations for language model compression. // *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, P 498–508, Online, November. Association for Computational Linguistics.

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

Raphael Tang, Yao Lu, Linqing Liu, Lili Mou, Olga Vechtomova, and Jimmy Lin. 2019. Distilling task-specific knowledge from bert into simple neural networks. *arXiv preprint arXiv:1903.12136*.

Yonglong Tian, Dilip Krishnan, and Phillip Isola. 2020. Contrastive representation distillation. // *International Conference on Learning Representations*.

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. // I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

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. // *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, P 5797–5808, Florence, Italy, July. Association for Computational Linguistics.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. SuperGLUE: A stickier benchmark for general-purpose language understanding systems. *arXiv preprint 1905.00537*.

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. // H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, *Advances in Neural Information Processing Systems*, volume 33, P 5776–5788. Curran Associates, Inc.

Wenhui Wang, Hangbo Bao, Shaohan Huang, Li Dong, and Furu Wei. 2021. MiniLMv2: Multi-head self-attention relation distillation for compressing pretrained transformers. // *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, P 2140–2151, Online, August. Association for Computational Linguistics.

John Wieting and Douwe Kiela. 2019. No training required: Exploring random encoders for sentence classification. // *International Conference on Learning Representations*.

Sanqiang Zhao, Raghav Gupta, Yang Song, and Denny Zhou. 2021. Extremely small BERT models from mixed-vocabulary training. // *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, P 2753–2759, Online, April. Association for Computational Linguistics.

A Models

We measured inference time and memory required for models from the Table 3 on NVIDIA GeForce GTX 1080 Ti and Intel(R) Core(TM) i7-7700K CPU @ 4.20GHz using benchmark utils from Transformers library[11]. For testing random sequences with `batch_size` $= 16$ and `sequence_length` $= 512$ were generated. We run each model 100 times to reduce an effect of possible external factors on time and memory values. Some of the distilled models require more memory for inference due to different implementations of DistilBERT and BERT architectures.

Model	# layers	# vocab, K	# hid	Params, M	Mem, MB	Inference time, ms		Inference mem, MB	
						cpu	gpu	cpu	gpu
teacher	12	119.5	768	177.9	679	5283.2	186.6	1550	1938
distil-base	6			135.5	517	2335.4	55.3	2177	2794
distil-small	2			**107.1**	**409**	**802.4**	**1.5**	**1541**	**1810**
distil-tiny30	3	30.5	264	10.4	41	374.7	2	714	1158
distil-tiny20		20		7.6	30	357.6	1.9	695	1148
distil-tiny10		10		5	19	356.5	**1.8**	679	1138
distil-tiny5		5		**3.6**	**14**	**354.9**	**1.8**	**664**	**1126**
rubert-tiny	3	29.6	312	**11.8**	**45.5**	**942.9**	**2.2**	**1308**	**2088**
rubert-tiny2		83.8		29.3	112	1786.6	2.3	3054	3848

Table 3: Teacher and student models characteristics. All models have 12 attention heads. "Mem" column is memory on disk required to store model, while "Inference time"/"Inference mem" is time/memory required for model to make inference on a given batch. Inference tests were made on batches of 16 random sequences with length 512. For distil-tiny* models, * corresponds to a vocabulary size in thousands.

Comparing to the teacher `distil-base` $1.3\times$ lighter and $3.5\times$ faster on GPU. At the same time `distil-small` is $1.7\times$ lighter and $126\times$ faster on GPU. But the memory required for inference remains almost the same as for teacher.

As vocabulary size decreases, the students `distil-tiny` are getting lighter: from $17\times$ to $49\times$ for models from 30k to 5k vocabulary. Inference time and memory holds almost the same order. Models `distil-tiny` are up to $104\times$ faster on GPU; memory consumption is up to 1.7 times lower on GPU. But still distil-small is the fastest of all students because of the lowest number of Transformer layers.

Nevertheless, rubert-tiny is $15\times$ lighter (`rubert-tiny2` $6\times$) than our teacher. Both models are $85\times$ faster on GPU, but require even more memory for inference.

B Training details

Our code is based on DistilBERT open-source implementation[12]. We trained students on 8 Tesla P100-SXM2-16Gb for 64 epochs with `batch_size` $= 4$, `gradient_accumulation_steps` $= 128$ and `AdamW` optimizer (Loshchilov and Hutter, 2017). For learning rate we applied warmup from 0 to $5e^{-4}$ and when required number of warmup steps passed, learning rate was halved after three validation epochs, if validation loss was not improved. We used DeepPavlov library (Burtsev et al., 2018) for our fine-tuning experiments.

B.1 Weights initialization

We initialized student models with parameters from the teacher. To initialize student embeddings we made the following steps:

1. Subwords from teacher vocabulary were split by student subwords (see *reduce* in Sec. 3.3).
2. For each student subword we collected corresponding teacher subwords in which that subword occured (according to the splits from previous step).

[11] huggingface.co/docs/transformers/benchmarks

[12] github.com/huggingface/transformers/tree/master/examples/research_projects/distillation

3. Student subword embeddings were initialized with averaged embeddings of the corresponding teacher subwords.

To initialize student layers, 12 Transformer layers of the teacher were grouped by 4 and averaged to match 3 student layers. Then we cut them to match student hidden states dimension.

B.2 Distilling teacher hidden states

The following steps were made:

1. Student and teacher model have different number of Transformer layers. Therefore, for each input token we averaged outputs of all Transformer layers for this token.
2. Match or reduce strategies were applied to align student sequence length dimension.
3. Averaged and aligned student hidden states were projected by fully-connected layer to match the teacher hidden states dimension. We initialized projection layers randomly (He et al., 2015) and use them in two modes – *frozen* and *trainable*.
4. MSE loss computed between aligned student and teacher hidden states.

C Experiments with different vocabulary sizes

As vocabulary size decreased, we expected more teacher knowledge would be lost, and students quality would decrease proportionally. Surprisingly we do not see this effect. For the same combination of losses `KL-match & MLM` we observe two groups of results in Table 4: 1. Scores on ParaPhraser and SberSQuAD increase as vocabulary size decreases. 2. Scores on RuSentiment and Collection-3 decrease as vocabulary become smaller.

Model	Proj	Distillation Losses	ParaPhraser	RuSentiment	Collection-3	SberQuAD	
			F1	F1 (weighted)	Entity F1	F1	EM
teacher	-	MLM, NSP	**86.30**±0.96	**76.00**±0.53	**97.01**± 0.13	**83.82**±0.15	**65.60**±0.12
distil-base	-	MLM, KL, MSE, Cos	82.86±0.47	75.82±0.98	96.40±0.20	80.05±0.43	60.96±0.51
distil-small	-		75.53±1.03	74.58±0.10	94.20±0.20	68.92±0.30	48.21±0.39
distil-tiny30	M	MLM, KL	74.59±0.20	**72.90**±0.20	**93.19**±0.17	52.64±0.37	34.74±0.41
distil-tiny20			74.35±0.59	72.49±0.21	92.57±0.15	48.46±1.39	31.11±1.34
distil-tiny10			74.58±0.24	72.50±0.24	92.20±0.14	64.05±0.82	44.66±0.83
distil-tiny5			**74.88**±0.33	70.86± 0.29	91.43±0.15	**67.46**±0.26	**47.82**±0.26

Table 4: Results for students with different vocabulary sizes. Teacher, distil-base, distil-small have 120k tokens in vocabulary.

D Fine-tuning datasets

ParaPhraser is a set of sentence pairs collected from news headlines and annotated as precise paraphrase, near paraphrase and non-paraphrase. The task we solve is binary classification – predict whether sentence pairs are paraphrases (precise or near paraphrases) or not. RuSentiment is a dataset for sentiment analysis of public posts on Russian social network VKontakte. Five categories were annotated "Neutral", "Negative", "Positive", "Speech Act", and "Skip". SberQuAD is a Russian QA dataset for a reading comprehension evaluation which contains paragraph–question–answer triples. Questions were constructed in such a way that answer is a some paragraph span. For NER task we used Collection-3: Persons-1000 collection[13] which contains names of persons, additionally annotated with organizations and locations named entities.

RussianGLUE is an advanced Russian general language understanding evaluation benchmark that contains nine tasks, collected and organized similarly to the SuperGLUE (Wang et al., 2019) methodology. The benchmark can be divided into six groups including the general diagnostics of language models, common sense understanding, natural language inference, reasoning, machine reading and world knowledge.

[13] ai-center.botik.ru/Airec/index.php/ru/collections/28-persons-1000

Dataset	Type	Metric	Train	Validation	Test
ParaPhraser	Classification	F1	6702	500	1899
RuSentiment		F1 (weighted)	31030	3448	4961
SberQuAD	Span prediction	F1, EM	45328	5036	-
Collection-3	NER	Entity F1	9301	2153	1922
		Russian SuperGLUE			
RUSSE	Common Sense	Acc	19845	8508	18892
PARus			500	100	400
TERRa	NLI	Acc	2616	307	3198
RCB		F1, Acc.	438	220	438
LiDiRus		MCC	0	0	1104
RWSD	Reasoning	Acc	606	204	154
MuSeRC	Machine Reading	F1, EM	500	100	322
RuCoS			72193	7577	7257
DaNetQA	World Knowledge	Acc	1749	821	805

Table 5: Summary of the common benchmark datasets for Russian with train/validation/test split sizes.