COMPRESSING BERT 25 TIMES BY RNN IN NAMED ENTITY RECOGNITION TASK

Ilya Malakhov
Novosibirsk State University
Russia, Novosibirsk
i.malakhov1@g.nsu.ru

Abstract

In named entity recognition task state-of-the-art models usually have a lot of parameters. It may become an obstacle to use these models on devices with limited performance due to big size and long inference time. In this work we used a pre-trained BERT-based model to train several RNN models with significantly less number of parameters using knowledge distillation technique. It was shown that such approach allows to achieve much better quality than usual training on a labeled task data. Also we proposed a data sampling approach and a specific loss function for data with imbalanced classes.

Keywords: named entity recognition, knowledge distillation, recurrent neural network, LSTM, SRU, BERT

1 Introduction

Today, Transformer-based architectures became a standard in the most of NLP tasks. But usually increasing prediction quality claims significant model's size growth. This makes limitations on using such models on mobile and other low performance devices. Here comes knowledge distillation technique that tends to teach some smaller model to imitate large accurate model's outputs. The model used as a teacher in this work is built based on BERT (Devlin et al., 2018) and has 85 million parameters. In our work we constructed several RNN (recurrent neural network) models with significantly less number of parameters ($\approx$5 million) and compared prediction quality of these models trained in two ways: distilling BERT with further fine-tuning vs usual straight training on the labeled data.

One of the advantages of recurrent models over Transformer-based architectures is that they can process sequences with unlimited length. As a result, we don't need to split too long sentences into several parts, in contrast to BERT if they exceed BERT maximum sequence length, so we will not lose context information over the sequence. One more reason why RNN architectures were chosen is the recent release of SRU++ (Lei, 2021) architecture that implements "fast recurrence" with an attention mechanism.

RuNNE (Russian Nested Named Entities) competition (Artemova et al., 2022) proposed to solve a NER (named entity recognition) task on a subset of NEREL (Loukachevitch et al., 2021) dataset (Russian news articles dataset, that contains annotated nested named entities). There were submitted several models by participants that formed a leaderboard based on F1 score averaged over entity types (F1-macro). In present work we have used RuNNE dataset and the third model from the competition leaderboard as a teacher that has BERT-based architecture.

2 Related works

Researchers interest in knowledge distillation (KD) grows with the models complexity. Bucila et al. (2006) showed that it is possible to compress ensemble of models into single model without significant loss in quality. Hinton et al. (2015) developed this further and showed that approximation large model's classes probabilities prediction in classification task is equivalent to minimizing MSE (mean squared error) between teacher and student output logits. So we used MSE for distillation training. Also they have used additional trick named "softmax-temperature" (T), but since our models is significantly larger it is hard to find such hyper-parameter so we didn't use it (T=1).

Chen et al. (2017) investigates on KD in object detection task using convolutional neural networks (CNN) and proposes using a one stage training using combined loss: classification loss plus loss measuring distance between teacher and student output. In our work we decided not to use this approach due to small dataset size and have split training in two stages.

Sanh et al. (2019) and Turc et al. (2019) proposed using KD training technique for BERT to significantly reduce number of parameters. For student they are using the same architecture as a teacher but with reduced number of layers and their dimensions. We are trying to use similar approach for different teacher and student architectures and our best student model has less parameter than smallest model by Turc et al. (2019) (BERT-Tiny). Also Turc et al. (2019) showed that distillation approach usually outperforms common training on labeled set.

3 Methodology

Our knowledge distillation approach is as follows: approximate teacher model's output distribution by student models through minimizing mean squared error (MSE) on a big dataset of unlabeled texts, then fine-tune students on a labeled data. Also we tried to train models without distillation step and compared achieved results.

Datasets. The main dataset in this work is the above mentioned **RuNNE** dataset, that contains 29 different entity types (Table 2). It was used for NER training and quality evaluation based on F1-macro score. But since it is too small ($\approx$5k sentences in train part after splitting into sentences), for BERT approximation we decided to take larger dataset. For that we used raw texts from **Gazeta** (Gusev, 2020) dataset that also consists of Russian news articles. Due to a large size of full Gazeta, only test part of it was used ($\approx$220k sentences after splitting into sentences).

Texts from each dataset were split into sentences by razdel sentenizer (Kukushkin, 2018) and then tokenized using WordPiece-based BertTokenizer from HuggingFace transformers library (Wolf et al., 2020). Than each token in RuNNE dataset was labeled with 29 BILOU (Beginning, Inside, Last, Outside, Unit) tags (one tag for each entity type) based on annotated entities. So we have 29 sequence tagging tasks.

Teacher. As was mentioned above we are using BERT-based model (further BERT) from RuNNE competition leaderboard that achieved 0.743 F1-macro score on the main task. This score doesn't pay attention to "few-shot" entity types (with too small occurrences). We have used F1-macro score averaged over all entity types (including "few-shot") types as target metric. So we measured quality of the teacher by our metric and got 0.711.

Students. Our main goal is to try mimicking BERT model by more lightweight RNN-based models. For this purpose we have built 3 models. As we are approximating BERT outputs each model uses same tokenization method and embeddings layer as BERT. Each model consists of three sequential parts: "conv" part, "rnn" part and "ner heads". "Conv" part and "ner heads" are identical for each model. "Conv" is two layers of stacked parallel 3 convolutions with kernel sizes 1, 3 and 5 and ReLU activation with BatchNorm (Ioffe and Szegedy, 2015). At the last convolution layer we reduce hidden vector size from 768 to ($384 = 768/2$) to reduce number of parameters. "Ner heads" are just 29 (for each entity type) parallel linear fully connected layers with output vector of size 5 for 5 BILOU tags. Models differ in "rnn" part:

- the first model has 4 bidirectional LSTM layers (Hochreiter and Schmidhuber, 1997) (further CLSTM model)
- the second model has 4 bidirectional SRU layers (Lei et al., 2017) (further CSRU model)
- the third model has 4 bidirectional SRU++ layers (Lei, 2021) (further CSRU++ model)

After each layer except the last one we applied dropout (Srivastava et al., 2014) with 0.1 rate. Models architecture is graphically presented at Figure 1.

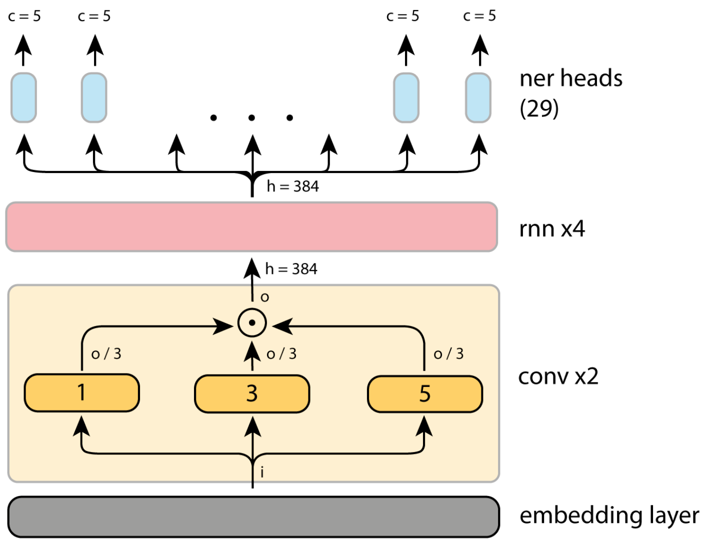

Figure 1: RNN-based student models architecture. Here i, o, h and c are hidden vector sizes for each token at specified stage. For the first conv layer $i = o = 768$. For the second conv layer $i = 768, o = 384$.

3.1 BERT distillation on Gazeta

At the first stage each model was trained to minimize **MSE** loss between self outputs (logits) and BERT outputs on Gazeta dataset using **AdamW** (Loshchilov and Hutter, 2017) optimizer with default betas, 0.01 weight decay and initial learning rate = $1\text{e}{-2}$ decreasing up to $1\text{e}{-4}$ through multiplying by 0.5 when F1-macro score stops increasing on RuNNE test dataset.

3.2 Fine tuning on RuNNE

Because of the imbalanced number of each entity type in the dataset, we need to make some tricks to balance prediction quality for each entity type. For this we made 2 things:

- Due to imbalanced entity types (see Table 2) we have made special sampler for training data that selects samples so that entities number of each type is proportional to logarithm of a total entities number of this type in dataset. At each step sampler remembers number of already sampled entities of each type and tries to select sample containing underrepresented entity type.
- Due to imbalanced BILOU tags we used specific loss function. For each sample it looks like:

$$Loss = \sum_{e=1}^{E} 0.2 * CE_e + DL_e \tag{1}$$

where $E = 29$ (entity types), CE_e stands for Cross-Entropy loss for e entity type with mean reduction over sequence and DL_e stands for Dice Loss. Also we tried to use self-adjusting technique showed by (Li et al., 2020), but it made training more unstable and gave worse results that without. So DL_e looks like:

$$
\begin{aligned}
DL_e &= \frac{1}{C}\sum_{c=1}^{C} w_c \frac{\gamma + 2\sum_{l=1}^{L} p_{lec} * y_{lec}}{\gamma + \sum_{l=1}^{L} p_{lec}^2 + y_{lec}^2} \\
p_{lec} &= \frac{\exp x_{lec}}{\sum_{c=1}^{C} \exp x_{lec}}
\end{aligned} \tag{2}
$$

where L is a sequence length, $C = 5$ (BILOU classes), x_{lec} – model's output at l position in sequence for e entity type and c BILOU tag, $y_{lec} = 1$ if real token at position l is labeled as e entity with c BILOU tag else 0, $w_c = 0$ for O tag and $w_c = 1$ for other tags, $\gamma = 1\text{e}{-6}$ for numerical stability. The final mini-batch loss is an average of the samples loss. Used optimizer is the same as in subsection 3.1.

3.3 Viterbi decoding algorithm

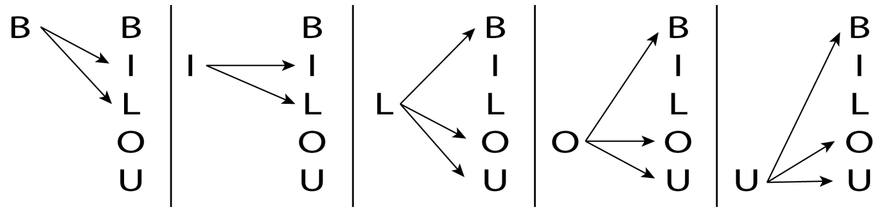

Figure 2: Possible BILOU tags transitions (B – beginning, I – inside, L – last, O – outside, U – unit).

Having predicted logits (model's outputs) we should decode them into final BILOU tags. One of the techniques for that is a Viterbi algorithm for decoding hidden Markov chains (Viterbi, 1967; Huang and Chiang, 2005). For each tag it is looking for a most likely state based on previous state and predefined state transition probabilities. In our case we have 5 states (BILOU). Probabilities for transitions that are not possible were defined as 0 and possible transitions have equal probabilities with 1 sum. For example from tag B we only can go into I or L and tags O and U can't appear here so their probabilities = 0. All possible transitions are shown at Figure 2. Having decoded BILOU tags we can get predicted entities and calculate result F1 score.

4 Results

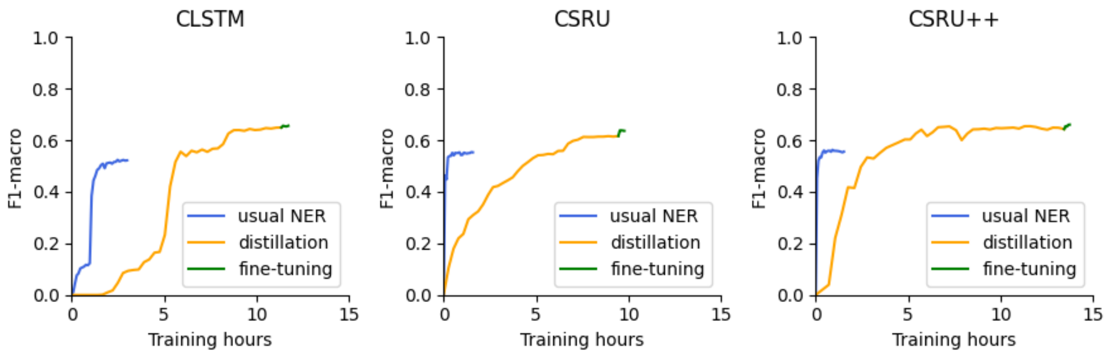

Figure 3: F1-macro score on test RuNNE dataset during training.

Eventually we have trained 6 RNN models: 3 different architectures in two ways as described in section 3. As far as "a picture is worth a thousand words" to be concise and not waste too much words all results are presented at Figure 3, Table 1 and Table 2.

Model	Number of parameters	Parameters reduce factor	CPU inference time (per sample)	GPU inference time (per sample)	F1 macro score (target metric)
BERT	85.5m	-	121ms	7ms	0.711
CSRU++ dist	3.4m	25.3	5ms	3ms	0.660
CLSTM dist	6.3m	13.7	12ms	4ms	0.656
CSRU dist	4.5m	19.1	6ms	3ms	0.636
CSRU++	3.4m	25.3	5ms	3ms	0.555
CSRU	4.5m	19.1	6ms	3ms	0.553
CLSTM	6.3m	13.7	12ms	4ms	0.522

Table 1: Results comparison for BERT and RNN models trained distilling BERT then fine-tuned as usual NER (with "dist" suffix) and trained without distillation (parameters in millions, time in milliseconds). CPU is a one core of Xeon Gold 6248. GPU is a one NVIDIA Tesla V100 SXM2 32GB.

Entity type	Number in dataset (sorted) (both train and test splits)	F1 usual NER by CSRU++	F1 distilled NER by CSRU++	F1 by BERT
F1-macro	5480	0.556	0.681	0.711
F1-main	5480	0.599	0.679	0.74
F1-few-shot	5480	0.177	0.502	0.456
PROFESSION	5480	0.646	0.681	0.715
PERSON	5480	0.841	0.886	0.968
ORGANIZATION	4744	0.619	0.706	0.751
EVENT	3600	0.466	0.556	0.599
COUNTRY	2977	0.934	0.95	0.958
DATE	2807	0.838	0.87	0.88
CITY	1342	0.798	0.853	0.938
NUMBER	1256	0.757	0.833	0.826
AGE	692	0.901	0.912	0.921
ORDINAL	672	0.798	0.795	0.858
NATIONALITY	460	0.688	0.698	0.825
LAW	458	0.352	0.444	0.476
AWARD	457	0.432	0.545	0.673
STATE_OR_PROVINCE	455	0.769	0.818	0.907
FACILITY	434	0.297	0.403	0.641
IDEOLOGY	343	0.605	0.692	0.722
LOCATION	336	0.5	0.598	0.579
PRODUCT	291	0.485	0.684	0.647
CRIME	217	0.424	0.539	0.701
MONEY	214	0.69	0.759	0.736
TIME	201	0.568	0.739	0.736
WORK_OF_ART	127	0.058	0.535	0.607
DISTRICT	123	0.146	0.632	0.756
RELIGION	118	0.75	0.744	0.936
PERCENT	89	0.2	0.471	0.615
DISEASE	89	0.615	0.615	0.427
PENALTY	51	0.667	0.571	0.333
LANGUAGE	51	0.273	0.5	0.667
FAMILY	31	0	0.125	0.211

Table 2: Detailed F1 scores for each entity type. F1-few-shot - averaged F1 over DISEASE, PENALTY and WORK_OF_ART (RuNNE competition metric). F1-main - averaged F1 over other entity types (RuNNE competition metric).

As we can see from Figure 3 SRU-based models was trained and converged faster than LSTM. Despite CSRU++ converged by F1-macro score at 7.5 hours of training, MSE loss on

Gazeta continued decreasing so we kept training before MSE has converged. After distillation fine-tuning stage gave a little increase in F1 score (0.01 – 0.03).

As was expected models that were pre-trained distilling BERT showed much better results than models trained without distillation technique. All models have significantly less parameters than BERT (CLSTM 13 times less, CSRU 19 times less and CSRU++ 25 times less) and are faster than big BERT model. Since idea of SRU architecture is "fast recurrence", CSRU and CSRU++ outperforms CLSTM by inference time. CSRU++ produced a little bit better score in our training conditions.

5 Discussion

Knowledge distillation is a useful technique for compressing a model. Parameters reduce factor inflicts the quality of the result model. Though we have lost 0.05 score points (from 0.71 for BERT to 0.66 for CSRU++) we managed to crucially reduce number of model parameters (25 times for CSRU++). We have carried out a number of the experiments with different hyperparameters but there may be some more suitable architectures and training tricks that might produce better results even with such reduce factor. We think that the first of all deserving attention to improve is a loss function. There may be helpful to use BILOU weights with Cross-Entropy loss or even different loss concept. We have investigated on using weighted CE a bit, but got worse results. Definitely other models architectures may be used to experiment with.

Our experiments scripts may be found at https://github.com/iluvvatar/bert-vs-rnn-runne.

Acknowledgements

Thanks to my scientific director Ivan Bondarenko (NSU, Russia Novosibirsk, i.bondarenko@g.nsu.ru) for providing BERT-based model used in RuNNE competition and consulting in some aspects of my work.

References

Ekaterina Artemova, Maksim Zmeev, Natalia Loukachevitch, Igor Rozhkov, Tatiana Batura, Pavel Braslavski, Vladimir Ivanov, and Elena Tutubalina. 2022. Runne-2022 shared task: Recognizing nested named entities. *Computational Linguistics and Intellectual Technologies: Proceedings of the International Conference "Dialog"*.

Cristian Bucila, Rich Caruana, and Alexandru Niculescu-Mizil. 2006. Model compression. // *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '06, P 535–541, New York, NY, USA. Association for Computing Machinery.

Guobin Chen, Wongun Choi, Xiang Yu, Tony X. Han, and Manmohan Chandraker. 2017. Learning efficient object detection models with knowledge distillation. // *NIPS*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding.

Ilya Gusev. 2020. Dataset for automatic summarization of russian news. // *Artificial Intelligence and Natural Language*, P 122–134, Cham. Springer International Publishing.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Liang Huang and David Chiang. 2005. Better k-best parsing. // *Proceedings of the Ninth International Workshop on Parsing Technology*, P 53–64, Vancouver, British Columbia, oct. Association for Computational Linguistics.

Sergey Ioffe and Christian Szegedy. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *CoRR*, abs/1502.03167.

Alexander Kukushkin. 2018. Rule-based token, sentence segmentation for russian language. `https://github.com/natasha/razdel`.

Tao Lei, Yu Zhang, Sida I. Wang, Hui Dai, and Yoav Artzi. 2017. Simple recurrent units for highly parallelizable recurrence.

Tao Lei. 2021. When attention meets fast recurrence: Training language models with reduced compute. *CoRR*, abs/2102.12459.

Xiaoya Li, Xiaofei Sun, Yuxian Meng, Junjun Liang, Fei Wu, and Jiwei Li. 2020. Dice loss for data-imbalanced nlp tasks. // *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, P 465–476.

Ilya Loshchilov and Frank Hutter. 2017. Fixing weight decay regularization in adam. *CoRR*, abs/1711.05101.

Natalia V. Loukachevitch, Ekaterina Artemova, Tatiana Batura, Pavel Braslavski, Ilia Denisov, Vladimir Ivanov, Suresh Manandhar, Alexander Pugachev, and Elena Tutubalina. 2021. Nerel: A russian dataset with nested named entities, relations and events. // *Proceedings of the International Conference Recent Advances in Natural Language Processing, RANLP 2021*, P 876–885, Online. Incoma Ltd.

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.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15:1929–1958, 06.

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

A. Viterbi. 1967. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Transactions on Information Theory*, 13(2):260–269.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Perric Cistac, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-Art Natural Language Processing. P 38–45. Association for Computational Linguistics, 10.