Computational Linguistics and Intellectual Technologies:
Proceedings of the International Conference "Dialogue 2025"

April 23–25, 2025

Methods for Recognizing Nested Terms

Igor Rozhkov
Lomonosov Moscow State University
Leninskie Gory, 1/4, Moscow, Russia
fulstocky@gmail.com

Natalia Loukachevitch
Lomonosov Moscow State University
Leninskie Gory, 1/4, Moscow, Russia
louk_nat@mail.ru

Abstract

In this paper, we describe our participation in the RuTermEval competition devoted to extracting nested terms. We apply the Binder model, which was previously successfully applied to the recognition of nested named entities, to extract nested terms. We obtained the best results of term recognition in all three tracks of the RuTermEval competition. In addition, we study the new task of recognition of nested terms from flat training data annotated with terms without nestedness. We can conclude that several approaches we proposed in this work are viable enough to retrieve nested terms effectively without nested labeling of them.

Keywords: nested terms, automatic term extraction, contrastive learning
DOI: 10.28995/2075-7182-2025-23-XX-XX

Методы распознавания вложенных терминов

Рожков И.С.
МГУ имени М.В. Ломоносова
Ленинские горы, 1/4, Москва, Россия
fulstocky@gmail.com

Лукашевич Н.В.
МГУ имени М.В. Ломоносова
Ленинские горы, 1/4, Москва, Россия
louk_nat@mail.ru

Аннотация

В этой статье мы описываем наше участие в конкурсе RuTermEval, посвященном извлечению вложенных терминов. Мы применяли модель Binder, которая ранее успешно применялась для распознавания вложенных именованных сущностей, для извлечения вложенных терминов. Мы получили наилучшие результаты распознавания терминов во всех трех треках конкурса RuTermEval. Кроме того, мы исследуем новую задачу распознавания вложенных терминов из плоских обучающих данных, аннотированных терминами без вложенности. Несколько методов, предложенных нами в данной работе, показывают, что вложенные термины возможно извлекать эффективно без наличия их вложенной разметки.

Ключевые слова: вложенные термины, автоматическое извлечение терминов, контрастивное обучение

1 Introduction

Automatic term extraction is a well-known research task that has been studied for decades. Terms are defined as words or phrases that denote concepts of a specific domain, and knowing them is important for domain analysis, machine translation, or domain-specific information retrieval. Various approaches have been proposed for automatic term extraction. However, automatic methods do not yet achieve the quality of manual term analysis.

During recent years, machine learning methods have been intensively studied (Loukachevitch, 2012; Charalampakis et al., 2016; Nadif and Role, 2021). The application of machine learning improves the quality of term extraction, but requires creating training datasets. In addition, the transfer of a trained model from one domain to another usually leads to degradation of the performance of term extraction. Currently, language models (Xie et al., 2022; Liu et al., 2020) are texted in automatic term extraction.

Available datasets usually contain so-called flat term annotation, that is, an annotated term cannot include other terms. Another formulation of the task assumes that terms can be contained within other terms, such terms are called nested terms. For example, the term "‘infection of upper tract" includes the following term: "infection", "tract", "upper tract". The extraction of nested terms allows the researcher to recognize more terms, but requires the application of special methods.

In 2024 new Russian datasets have been prepared for automatic term analysis in the framework of the RuTerm-2024 evaluation. The datasets are annotated with nested terms. The dataset includes texts from different domains, which allows for the study of model transfer between domains.

In this paper, we consider an approach for the extraction of nested terms and test it in the RuTerm-2024 dataset. In addition, we study the task of the transfer from flat terms to nested ones, that is, we suggest that the training data are annotated only with flat terms, but the trained model should can annotated nested terms. For both tasks, we experiment with the Binder model, which creates representations for target entity types using contrastive learning (Zhang et al., 2023).

2 Related Work

The automatic term extraction task has two main variants: corpus-oriented and document-oriented (Šajatović et al., 2019). In the former case, a system should generate a list of terms for a document corpus; in the latter case, the task is to identify mentions of terms in domain documents. This variant can be classified as a sequence labeling task.

The first corpus-oriented methods utilized statistical measures such as tf-idf, context measures (c-vaule), association measures based on frequencies of component words and co-occurrence frequencies of a candidate phrase (Astrakhantsev et al., 2015). The combination of measures leads to a significant improvement in term extraction performance (Bolshakova et al., 2013; Loukachevitch, 2012), but the transfer of a model trained for one domain to another domain leads to degradation of results (Hazem et al., 2022; Loukachevitch and Nokel, 2013; Bolshakova and Semak,).

In document-oriented term extraction, the authors of (Bolshakova and Efremova, 2015) exploited lexico-syntactic patterns and rules. Currently, the most effective approaches to document-oriented term extraction utilized BERT-based techniques (Lang et al., 2021; Tran et al., 2024).

The extraction of nested terms was studied in several works (Marciniak and Mykowiecka, 2015; Tran et al., 2024; Vo et al., 2022). The authors of (Marciniak and Mykowiecka, 2015) divide longer terms into syntactically correct component phrases and estimate the "unithood" of candidate terms using the NPMI statistical measure (Normalized Pointwise Mutual Information) (Bouma, 2009). Tran et al. (Tran et al., 2024) propose using a novel term annotation scheme for training models. The proposed scheme comprises an additional encoding for nested single terms. Applying the scheme and XLMR classifier, the authors obtain the best results on the ACTER multilingual dataset (Rigouts Terryn et al., 2022).

In a similar task of recognition of nested named entities, various approaches have been proposed (Loukachevitch et al., 2024; Zhang et al., 2023; Zhu et al., 2022a). One of the best methods for nested named entity recognition is Binder (Zhang et al., 2023), based on contrastive learning.

3 RuTermEval Competition and Dataset

As part of the Dialogue Evaluation in 2025, the RuTermEval competition was organized. In this competition, one should design a model capable of extracting nested terms given the labeled data.

There are three tracks in this competition:

- Track 1: Identification of terms
- Track 2: Identification and classification of terms into three classes (specific_term, common_term, nomen);
- Track 3: Identification and classification of terms into three classes (specific_term, common_term, nomen) with the formulation of the transfer learning task to other domains.

The organizers used Codalab to hold the competition. In the first track they used the usual F1 score for measuring the teams' submissions accuracy. For the second and third tracks, organizers used weighted F1 and class-agnostic F1 scores. The weighted F1 considers some classes more important than the others,

	train	dev (track 1 & 2)	dev (track 3)
specific	12664 (69.95%)	3387 (67.77%)	3270 (58.52%)
common	4866 (26.88%)	1275 (25.51%)	1173 (20.49%)
nomen	573 (3.17%)	336 (6.72%)	1145 (20.99%)
total	**18103**	**4998**	**5588**

Table 1: Dataset term class amount and relative count.

nestedness	class	character length			word length		
		min	max	mean	min	max	mean
outermost	specific	2	91	17.58	1	13	1.89
	common	2	41	10.98	1	10	1.36
	nomen	3	95	24.06	1	14	3.19
	all	2	95	17.58	1	14	1.84
inner	specific	1	83	13.95	1	13	1.46
	common	2	31	7.54	1	5	1.12
	nomen	3	11	7.11	1	3	1.22
	all	1	83	13.95	1	13	1.32
overall	specific	1	91	17.83	1	13	1.79
	common	2	41	9.47	1	10	1.25
	nomen	3	95	23.79	1	14	3.16
	all	1	95	15.78	1	14	1.69

Table 2: Train dataset terms length.

thus increasing their value ratio in the total score. The second one treats the task of term extracting as term identification task, i.e. measuring the accuracy of term prediction regardless of their class.

The total table of competitors' results was sorted based on the F1 score in the first track and the weighted F1 score in the second and third tracks.

Three types of terms were considered:

- *specific*: terms that are specific both in-domain and lexically;
- *common*: terms that are specific only in-domain (they can be known and used by non-specialists);
- *nomen*: names of unique objects that belong to a specific domain.

Examples for each of three term types are as follows:

(1) *эпистемической* *модальности*
epistemic modality
'Example of specific term'

(2) *пользователю*
user
'Example of common term'

(3) *Национального* *корпуса* *русского* *языка*
National Corpus of Russian Language
'Example of nomen term'

In Table 1, the frequencies of each term type in each track are shown. It can be seen that specific terms are the most frequent in the dataset. Table 2 demonstrate the length distribution for each term type in the training set. We can see that nomen terms are longest (their length is about 3 words on average), and common terms are shortest (1-2 words on average).

level	specific	common	nomen	total
1 (outermost)	9589 (75.72%)	2734 (39.07%)	564 (96.91%)	12887 (55.26%)
2	2690 (21.24%)	1734 (24.78%)	9 (1.55%)	4433 (19.01%)
3	360 (2.84%)	367 (5.24%)	0 (0%)	727 (3.12%)
4	25 (0.20%)	30 (0.43%)	0 (0%)	55 (0.24%)
5	0 (0%)	1 (0.01%)	0 (0%)	1 (0.01%)
total (inner)	3075 (19.54%)	2132 (30.47%)	9 (1.55%)	5216 (22.37%)
total (overall)	15739	6998	582	23319

Table 3: Nestedness count of train dataset

level	specific	common	nomen	total
1 (outermost)	2606 (62.52%)	749 (41.59%)	269 (66.75%)	3624 (56.87%)
2	668 (16.03%)	423 (23.49%)	67 (16.63%)	1158 (18.17%)
3	92 (2.21%)	87 (4.83%)	0 (0%)	179 (2.81%)
4	18 (0.43%)	12 (0.67%)	0 (0%)	30 (0.47%)
5	3 (0.07%)	4 (0.22%)	0 (0%)	7 (0.11%)
total (inner)	781 (18.74%)	526 (29.21%)	67 (16.63%)	1374 (21.56%)
total (overall)	4168	1801	582	6372

Table 4: Nestedness count of dev dataset (track 1 & 2)

In Tables 3, 4, 5, 6, the levels of term nestedness for different sets and tracks are shown. The maximal level of nestedness in term annotations is 5. The fraction of longest (outmost) entities is about 55-56%. That is, nested terms constitute a significant share of annotated terms.

4 RuTermEval Competition Solution

4.1 Nested Named Entity Extraction

We depicted the term extraction task of the RuTermEval competition as a named entity extraction task, since both tasks are sequence labeling tasks. Moreover, the term extraction task could be framed as a named entity extraction task on a specific domain, that is, terms in the RuTermEval dataset. Thus, terms are onwards considered entities throughout this paper.

More specifically, in entity recognition tasks, for a given text sequence $X = \{x_1, x_2, \ldots, x_n\}$, where n is its length, we need to assign each subsequence $E = \{(x_i, x_{i+1}, \ldots, x_j) \mid 1 \leqslant i \leqslant j \leqslant n; i, j, n \in \mathbb{N}\}$ to its corresponding label $y_{i,j} \in Y$ (if any), where Y is a predetermined set of entity classes.

In the usual setting of the "flat" entity recognition task $\forall e_1, e_2 \in E : e_1 \nsubseteq e_2$, i.e., entities in a sequence cannot contain other entities within them. Moreover, $\forall e_1, e_2 \in E : e_1 \cap e_2 = \varnothing, e_1 \neq e_2$ i.e., such entities do not overlap too. We will denote such labeling as F:

$$F = \{e \in E \mid \forall e_1, e_2 : e_1 \cap e_2 = \varnothing\}.$$

On the other hand, in practice, entities can overlap. In this case, we are having nested labeling, i.e. $\exists e_1, e_2 \in E : e_1 \subset e_2$. We will denote such data as N:

$$N = \{e \in E \mid \exists\, e_1, e_2 : e_1 \subset e_2; \forall\, e_3, e_4 : \text{either } e_3 \subset e_4 \text{ or } e_4 \subset e_3 \text{ or } e_3 \cap e_4 = \varnothing\}.$$

Note: $F \subset N$.

The RuTermEval competition is devoted to the recognition of nested terms. Our solution is based on the Binder model, learned on training data with the ruRoberta-large pretrained language model (Zmitrovich et al., 2023).

level	specific	common	nomen	total
1 (outermost)	2404 (58.12%)	705 (44.48%)	940 (66.86%)	4049 (56.81%)
2	722 (17.46%)	373 (23.53%)	215 (15.29%)	1310 (18.38%)
3	131 (3.17%)	57 (3.60%)	15 (1.07%)	203 (2.85%)
4	13 (0.31%)	10 (0.63%)	3 (0.21%)	26 (0.36%)
5	0 (0%)	0 (0%)	0 (0%)	0 (0%)
total (inner)	866 (20.94%)	440 (27.76%)	233 (16.57%)	1539 (21.59%)
total (overall)	4136	1585	1406	7127

Table 5: Nestedness count of dev dataset (track 3)

inner \ outer	specific	common	nomen
specific	3124	3	358
common	1508	773	281
nomen	4	0	5
total	4636	776	644

Table 6: Inner classes of terms inside outermost terms of train dataset

4.2 Binder Model

The main model used in the competition and in this study was the `Binder` model (Zhang et al., 2023). This model allows for extracting entities (named entities, terms) from sentences based on the so-called descriptions of entities and a contrastive learning method.

The input is the sequence of words $x_1, \ldots, x_n$ and the description of the named entity E_k of class k. Both are fed to the so-called encoders, which represent the original text data as vector representations (the so-called embeddings), enriched with contextual information about each word. The most popular encoder is a model such as `BERT` (Kenton and Toutanova, 2019). The `BERT` model also allows for obtaining vector representation of the whole sentence using a special token `[CLS]`.

Each obtained vector is mapped onto a single vector space using a linear layer. In this space, it is assumed that all entities of the same type should be closer to each other than any other entity, as well as any other subsequences of words of the original sequence.

The main idea of the model is so-called contrastive learning. It consists in the fact that the model learns not from the vector sequences of words and their subsequences but learns to bring entities of one type closer to a certain single center of the most characteristic entity of this type and, conversely, to move away all other uncharacteristic subsequences.

At the prediction stage, the model is required to find out where the boundary between positive and negative mentions is drawn. To do this, based on the `[CLS]` vector of the second encoder used to encode the main sequence, its position in the same vector space is trained, and the distance between it and the anchor is the desired radius within which the mentions are positive. This is the so-called dynamic threshold approach, since it is trained together with the rest of the model, and is not fixed in advance.

4.3 Model parameters

This solution was the same for all three tracks. In the first track, we considered all terms mentions as entities of the dummy class *any*, treating the task of term identification as term classification of a single entity type. In tracks 2 and 3, the Binder model was trained to extract three term types.

We trained this model on 128 epochs on GPU with batch size of 8, maximum sequence length of 192, doc_stride of 16, learning rate of $3 * 10^{-5}$ using AdamW optimizer (Loshchilov and Hutter, 2019) with default settings. These parameters were the same for the solutions for the three tracks.

User	Team Name	Scoreboard F1 score
fulstock (ours)	**LAIR RCC MSU**	**0.7940**
VladSemak	VSemak	0.7685
ivan_da_marya	Ivan da Marya	0.5619
ragunna	KiPL SPBU	0.5349
angyling	-	0.5333
VatolinAlexey	ai	0.0000

Table 7: All teams results on the Track 1 of RuTermEval competition.

User	Team Name	Total score	Weighted F1 score	Class-agnostic F1 score
fulstock (ours)	**LAIR RCC MSU**	**0.6997**	**0.6997**	**0.78**
VladSemak	VSemak	0.6996	0.6996	0.77
VatolinAlexey	ai	0.5797	0.5797	0.63
ragunna	KiPL SPBU	0.5043	0.5043	0.52
angyling	-	0.3137	0.3137	0.53

Table 8: All teams results on the Track 2 of RuTermEval competition.

In Tables 7, 8, 9 we show the results of our submissions, compared to results of other teams.

We can see that the approach of nested term recognition based on the Binder model came out first on all three tracks, overtaking other teams submissions. Though, in the second track the next participant got the score almost identical to ours — with negligibly small difference. Thus, we can consider that our method surpasses the others in first and third tracks, but stays on the same line of prediction performance as the second-best method in the second track.

4.4 Error analysis

We perform error analysis on the predictions of the development dataset.

First, we can see that the model confuses common and specific terms, for example:

(4) *оценке* *сочетаемости* *слов*
assessment of compatibility of words

(5) *синтаксического* *словаря*
syntactic dictionary

The model recognized these terms as common, though they were actually labeled as specific. However, the following examples were labeled common, but the model recognized them as specific.

(6) *повтору*
repeat

(7) *модальная*
modal

The confusion of these classes occurs rather frequently. We believe this is due to their definition: model fails to differ when the found term is used by non-specialists and when it is not. We can also see that in the first track there are no such errors.

Second, from our results, we see that most of the model prediction errors were false negative, i.e., the model retrieves less amount of erroneous terms than not retrieves the labeled ones.

Third, we see some labeling ambiguity in the original data. For example,

User	Team Name	Total score	Weighted F1 score	Class-agnostic F1 score
fulstock (ours)	**LAIR RCC MSU**	**0.4823**	**0.4823**	**0.60**
VladSemak	VSemak	0.4654	0.4654	0.51
angyling	-	0.4370	0.44	0.53

Table 9: All teams results on the Track 3 of RuTermEval competition.

(8) *ядро* *системы* *персонализированного* *синтеза* *речи* *по* *тексту*
core of system of personalized synthesis of speech from text

(9) *системы* *персонализированного* *синтеза* *речи*
system of personalized synthesis of speech

(10) *персонализированного* *синтеза* *речи*
personalized synthesis of speech

were all labeled as specific, but

(11) *системы* *персонализированного* *синтеза* *речи* *по* *тексту*
system of personalized synthesis of speech from text

was not. The model recognized all four of these examples as specific terms, but because the last was not labeled, it resulted in a false negative error.

5 Nested Term Recognition from Flat Supervision Task

Currently, most existing datasets labeled with terms have flat annotations. However, we can see that many terms can be nested. Therefore, we study methods for extracting nested terms from flat-term annotations based on the RuTermEval data.

5.1 Task

We consider the problem of extracting nested data given only flat labeling. That is, for training data only flat data F is given, but for validation and test subsets we have full nested data N. This task was proposed for named entities in (Zhu et al., 2022b).

There are different options available for flat data F. For example, if F is generated from nested data N by deleting all overlaps $e_1 \subset e_2, e_1, e_2 \in N$, we can delete only e_1 or e_2 from N to remove this overlap. Thus, we come to many scenarios of flat data.

We consider the ultimate and most common option — "outermost" flat data. By "outermost" we mean entities that can contain other entities, but are not part of any other entity in data, i.e., "outermost" flat data $F = F_u$ is as:

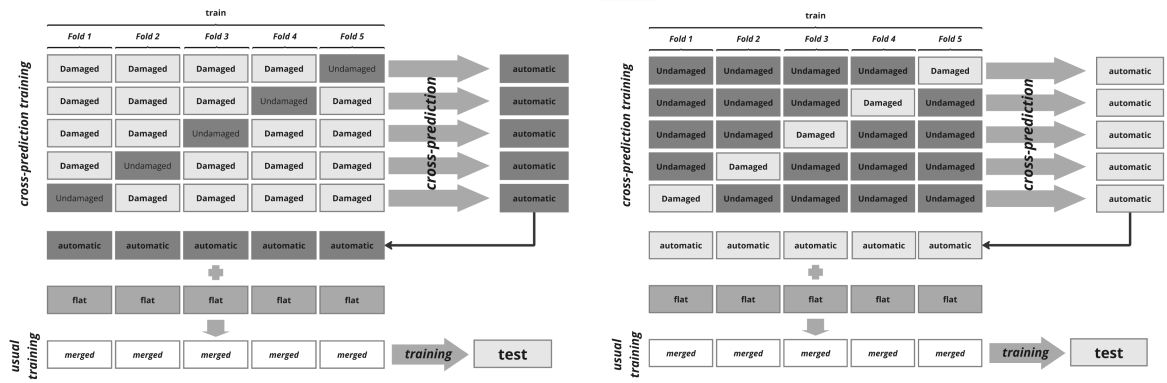
$$F_u = \{e \in N \mid \nexists e' : e \subset e'\}$$

In this work, we propose several methods for the prediction of nested data N given only "outermost" flat data F_u. Furthermore, we denote all remaining nested entities that are not part of outermost flat data as inner entities I: $I = N \setminus F_u$.

5.2 Baselines

Two baselines are used: full nested learning and "pure" flat learning.

Full nested learning is the option to learn a model on all available nested data N to achieve the best results in extracting nested entities as described in Section 4. This baseline would give us the upper limit that should be achieved through flat supervision methods.



(a) ”Early” damage

(b) ”Late” damage

Figure 1: Scheme of damaged cross-prediction method, example of 5 folds.

”Pure” flat learning is the option of learning only on available flat data F without any changes to the learning scheme, model architecture, or data augmentation. This is the simplest approach in flat supervision and is regarded as a bottom baseline.

Both approaches would give us the upper and lower expected limits.

5.3 Automatic pseudo-labeling

In this work, we propose several novel methods grouped by the term ”automatic pseudo-labeling”. All these approaches represent data augmentation techniques that try to leverage the omitted nestedness of the data.

There are four approaches in total, as follows:

- ”Inclusions” (simple and lemmatized);
- ”Damaged cross-validation” (”early” and ”late”).

The explanations are given below.

5.3.1 Inclusions

Consider the flat dataset F_u defined as above. Entities in such dataset can contain inner entities within them; they were either unlabeled or deleted from initial labeling. Since the given text itself was not altered, the entities are still hidden in the text, an we just cannot know where exactly.

In practice, many entities do not appear just once in the text. Each occurrence is called ”mention”, which is widely known in other fields such as entity linking. Furthermore, each entity can have many mentions in the text.

This phenomenon inspired the approach of so-called ”inclusions”. We consider the hypothesis that each mention of the known entities should be tackled as entities too. Inclusions are mentions of known entities inside other ones.

We add such inclusions to the initial flat dataset as a pseudo-labeling, much like data augmentation, and train model on a new dataset. Inclusions are computed only at the training phase.

Another variant of this method additionally exploits the lemmatization of such inclusions. Before we looked for mentions of known entities ”as whole,” i.e., each mention should be written exactly as the original entity itself. But if we split such entities as a list of words, lemmatize and combine them in one unordered set of lemmas, we can extend the definition of inclusions.

From our calculations, there were 1296 simple inclusions (and 14183 entities as a result) and 3681 lemmatized inclusions (and 16568 entities as a result) in the train dataset.

Approaches	dev						test
	F1 micro, %			F1 macro, %			Scoreboard F1 %
	overall	inner	outer	overall	inner	outer	
pure flat	64.73	0.43	71.53	66.23	0.39	73.04	65.10
inclusions	68.28	17.78	**71.65**	67.79	11.74	71.92	67.42
lemm. inclusions	70.51	32.69	71.28	70.47	23.50	72.24	70.26
early damage	69.87	23.76	71.13	72.43	25.65	72.64	71.09
late damage	66.66	10.57	70.06	68.63	10.43	72.31	68.90
lemm. inc. + early dmg	**71.99**	**38.42**	70.52	**74.30**	**36.04**	73.18	**72.81**
lemm. inc. + late dmg	71.16	32.88	71.56	71.91	28.25	**73.26**	71.89
full	79.87	67.73	72.90	81.84	65.20	74.68	79.40

Table 10: Results on track 1.

Approaches	dev						test	
	F1 micro, %			F1 macro, %			w. F1, %	c/a F1, %
	overall	inner	outer	overall	inner	outer		
pure flat	61.01	0.68	67.04	63.61	1.42	69.62	55.04	63.60
inclusions	64.98	17.79	**67.96**	65.50	12.97	69.28	58.64	66.60
lemm. inclusions	66.84	30.80	67.55	67.02	24.06	68.60	59.49	70.21
early damage	65.34	19.34	65.32	66.96	20.86	66.97	60.39	70.03
late damage	63.18	8.63	66.68	64.51	8.68	68.17	58.84	67.29
lemm. inc. + early dmg	**68.64**	**36.40**	66.40	**69.52**	**32.15**	68.20	**63.10**	**73.37**
lemm. inc. + late dmg	67.13	32.97	66.66	68.31	28.07	**69.74**	60.97	71.41
full	76.06	64.76	68.44	77.40	61.29	70.41	69.97	77.79

Table 11: Results on track 2.

5.3.2 Damaged Cross-prediction

Another approach to pseudo-labeling guides the model during training to "look for" more inner entities.

Firstly, flat training data is divided into K folds (K is predefined). These folds are used for cross-validation: $K-1$ of them are used as a subtraining dataset, while the remaining fold is used as a subdevelopment dataset.

Secondly, we change the given flat data or "damage" it. Only training subset is changed. Here we propose two different methods:

- "Early" damage. We delete labeling of all long flat entities (of length 3 and more) from all subtraining folds, while changing one of the words x_t of such entity to some other character sequence.
- "Late" damage. Same as "early", but we damage the subdevelopment fold, while subtraining folds remain untackled.

From our calculations, out of 12887 of train subset entities, 2053 were damaged, while 10834 others were kept the same.

Thirdly, we train model K times, just like in cross-validation: $K-1$ subtraining folds are used for training, while the remaining fold is used for testing. For this purpose, in both methods the model tries to predict entities in parts of the initial training dataset. After all K training procedures, we get predictions of the model on all K folds. All these predictions are now considered as additional pseudo-labeling.

Finally, obtained labeling is added to the initial flat training subset. After that, the method stays familiar with usual training and evaluation. Figure 1 depicts the early damage and late damage pseudo-labeling schemes.

Approaches	dev						test	
	F1 micro, %			F1 macro, %			w. F1, %	c/a F1, %
	overall	inner	outer	overall	inner	outer		
pure flat	40.20	1.79	42.36	36.76	1.66	38.67	40.17	51.58
inclusions	43.99	9.99	44.36	39.78	7.78	40.61	42.05	55.54
lemm. inclusions	47.17	18.32	45.10	41.65	14.42	40.80	43.69	57.23
early damage	45.84	13.02	44.44	40.85	9.98	40.68	43.39	55.99
late damage	37.21	2.08	39.26	33.88	1.96	35.95	41.02	54.10
lemm. inc. + early dmg	**48.62**	**20.49**	**46.19**	**43.52**	**15.86**	**42.50**	**45.47**	**58.75**
lemm. inc. + late dmg	46.77	17.50	45.07	41.16	13.79	40.50	43.95	56.94
full	50.35	29.57	46.18	45.16	23.91	42.18	48.23	60.38

Table 12: Results on track 3.

All approaches were conducted in all three tracks, both on development and test subsets. Results are given in 10, 11, 12. In the RuTermEval competition in the second and third tracks two metrics were utilized namely weighted F1 (denoted as w. F1) and class-agnostic F1 (denoted as c/a F1).

First, we can see in all three tracks that pure flat approach extracts almost none of inner entities, while still capable of extracting outermost entities, though underperforming.

Second, as first two tracks consider same data but with different labeling, we see similar results on all approaches. But when task switches from term identification to term classification, all methods perform a bit poorer in general. Moreover, the third track considers the prediction on the different data domain — hence worse results in comparison.

Third, we can see that approaches on development and test sets share similar relative results.

Fourth, we see that all approaches enhance the performance of the inner terms prediction, hence the better result in the competition too.

Fifth, inclusions achieve good boost performance with its pseudo-labeling, from 0.68% to 17.79% inner terms prediction on second track. Moreover, lemmatized inclusions boost the results even more — up to 30.80%.

Sixth, damage cross-prediction methods achieve results around the same level of inclusions. We see that the "late" damage appeared to be poorer in general: model could not really retrieve damaged entities. We believe that this is due change of the context and hence embeddings confusion.

Seventh, merging together results of damaged cross-prediction and lemmatized inclusions, we achieved best inner terms predictions performance, up to 36.40%. On third track, this approach almost achieves the full nested approach, 58.75% to 60.38%. Thus, we see that such simple pseudo-labeling can be both very helpful and viable for recognizing nested terms from flat data.

5.4 Error analysis

As in our RuTermEval solution, we perform the same error analysis of our methods on predictions on the development dataset.

Firstly, we see that the model receives the ambiguity of the predictions because of nestedness. For example,

(12) *лексической* *системы* *языка*
lexical system of language
is labeled as specific and model did not recognize it at all, but

(13) *лексической* *системы*
lexical system
was recognized by model as specific term, though it was not labeled in data.

Secondly, we see another common pattern. Due to much less nested data in all our approaches with pseudo-labeling, model still fails to generalize on inner terms. Nevertheless, with our approach model now does recognize some flat terms that the usual nested approach of our best solution failed to extract instead:

(14) *озимой* *пшеницы*
winter wheat
was extracted correctly in flat approaches, while nested solution did not recognize it (though it was labeled);

(15) *пшеницы*
wheat

(16) *зерна*
grain
were mistakenly extracted as terms by flat approaches, though they were actually not (i.e., not labeled so).

We believe that the model got better trained on longer entities, which resulted in such prediction behaviour.

Still, this hypothesis holds true to erroneous predictions: for longer subsequences where there was no term, the model retrieved them in flat approaches, while the best nested solution did not:

(17) *биологического* *биоцидного* *препарата* *для* *борьбы* *с* *вредными* *членистоногими*
biological biocidal preparation for the control of harmful arthropods

was mistakenly extracted as a term in flat approaches. We see that such errors were the most frequent.

Thirdly, our pseudo-labeling inclusions approach introduced to the trained model that many mentions of some term are always terms too. Thus, there are many mistakes such as:

(18) *почва*
soil
at positions 239, 243; 901, 906; 1265, 1270; 1304, 1309, etc. in text track3-test1-81

(19) *урожайность*
productivity of land
at positions 132, 143; 298, 309; 437, 448; 567, 578; 641, 652, etc. in text track3-test1-81

while nested solution did not produce them.

6 Conclusion

In this work, we present our solution to the RuTermEval competition to all three tracks. Our solution via regarding Nested Term Extraction task as a Nested Named Entity Extraction task appeared to be the most effectively or at least alongside other competitors' solutions — we have got the first place in all three tracks, devoted to different scenarios of term extraction. We used the Binder model, which was previously successfully applied to the recognition of nested named entities, to extract nested terms and obtained the best results of term recognition in all three tracks of the RuTermEval competition.

We describe and motivate a new task — Nested Term Recognition from Flat Supervision. In this task, a model should predict nested terms based only on flat labeling at hand. We propose some approaches to this task and evaluate them in the RuTermEval competition. Of them, the combined lemmatized inclusions with early damage approaches resulted in the best inner term prediction score, coming close to the score on the full nested data.

References

Nikita A Astrakhantsev, Denis G Fedorenko, and D Yu Turdakov. 2015. Methods for automatic term recognition in domain-specific text collections: A survey. *Programming and Computer Software*, 41:336–349.

Elena I Bolshakova and Natalia E Efremova. 2015. A heuristic strategy for extracting terms from scientific texts. // *Analysis of Images, Social Networks and Texts: 4th International Conference, AIST 2015, Yekaterinburg, Russia, April 9–11, 2015, Revised Selected Papers 4*, P 297–307. Springer.

Elena I Bolshakova and Vladislav V Semak. An experimental study on cross-domain transformer-based term recognition for russian.

Elena Bolshakova, Natalia Loukachevitch, and Michael Nokel. 2013. Topic models can improve domain term extraction. // *Advances in Information Retrieval: 35th European Conference on IR Research, ECIR 2013, Moscow, Russia, March 24-27, 2013. Proceedings 35*, P 684–687. Springer.

Gerlof Bouma. 2009. Normalized (pointwise) mutual information in collocation extraction. *Proceedings of GSCL*, 30:31–40.

Basilis Charalampakis, Dimitris Spathis, Elias Kouslis, and Katia Kermanidis. 2016. A comparison between semi-supervised and supervised text mining techniques on detecting irony in greek political tweets. *Engineering Applications of Artificial Intelligence*, 51:50–57, May.

Amir Hazem, Mérieme Bouhandi, Florian Boudin, and Béatrice Daille. 2022. Cross-lingual and cross-domain transfer learning for automatic term extraction from low resource data. // *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, P 648–662.

Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. // *Proceedings of naacL-HLT*, volume 1. Minneapolis, Minnesota.

Christian Lang, Lennart Wachowiak, Barbara Heinisch, and Dagmar Gromann. 2021. Transforming term extraction: Transformer-based approaches to multilingual term extraction across domains. // *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, P 3607–3620.

Zhuang Liu, Degen Huang, Kaiyu Huang, Zhuang Li, and Jun Zhao. 2020. FinBERT: A Pre-trained Financial Language Representation Model for Financial Text Mining. // *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence*, P 4513–4519, Yokohama, Japan, July. International Joint Conferences on Artificial Intelligence Organization.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization.

Natalia Loukachevitch and Michael Nokel. 2013. An experimental study of term extraction for real information-retrieval thesauri. // *Proceedings of TIA*, P 69–76.

Natalia Loukachevitch, Ekaterina Artemova, Tatiana Batura, Pavel Braslavski, Vladimir Ivanov, Suresh Manandhar, Alexander Pugachev, Igor Rozhkov, Artem Shelmanov, Elena Tutubalina, et al. 2024. Nerel: a russian information extraction dataset with rich annotation for nested entities, relations, and wikidata entity links. *Language Resources and Evaluation*, 58(2):547–583.

Natalia V Loukachevitch. 2012. Automatic term recognition needs multiple evidence. // *LREC*, P 2401–2407.

Malgorzata Marciniak and Agnieszka Mykowiecka. 2015. Nested term recognition driven by word connection strength. *Terminology. International Journal of Theoretical and Applied Issues in Specialized Communication*, 21(2):180–204.

Mohamed Nadif and François Role. 2021. Unsupervised and self-supervised deep learning approaches for biomedical text mining. *Briefings in Bioinformatics*, 22(2):1592–1603, March.

Ayla Rigouts Terryn, Veronique Hoste, and Els Lefever. 2022. Acter 1.5: Annotated corpora for term extraction research. // *CLARIN Annual Conference Proceedings, 2022*, P 1–4.

Antonio Šajatović, Maja Buljan, Jan Šnajder, and Bojana Dalbelo Bašić. 2019. Evaluating automatic term extraction methods on individual documents. // *Proceedings of the Joint Workshop on Multiword Expressions and WordNet (MWE-WN 2019)*, P 149–154.

Hanh Thi Hong Tran, Matej Martinc, Andraz Repar, Nikola Ljubešić, Antoine Doucet, and Senja Pollak. 2024. Can cross-domain term extraction benefit from cross-lingual transfer and nested term labeling? *Machine Learning*, 113(7):4285–4314.

Chau Vo, Tru Cao, Ngoc Truong, Trung Ngo, and Dai Bui. 2022. Automatic medical term extraction from vietnamese clinical texts. *Terminology*, 28(2):299–327.

Qianqian Xie, Jennifer Amy Bishop, Prayag Tiwari, and Sophia Ananiadou. 2022. Pre-trained language models with domain knowledge for biomedical extractive summarization. *Knowledge-Based Systems*, 252:109460, September.

Sheng Zhang, Hao Cheng, Jianfeng Gao, and Hoifung Poon. 2023. Optimizing Bi-Encoder for Named Entity Recognition via Contrastive Learning, February.

Enwei Zhu, Yiyang Liu, Ming Jin, and Jinpeng Li. 2022a. Recognizing Nested Entities from Flat Supervision: A New NER Subtask, Feasibility and Challenges, November.

Enwei Zhu, Yiyang Liu, Ming Jin, and Jinpeng Li. 2022b. Recognizing nested entities from flat supervision: A new ner subtask, feasibility and challenges.

Dmitry Zmitrovich, Alexander Abramov, Andrey Kalmykov, Maria Tikhonova, Ekaterina Taktasheva, Danil Astafurov, Mark Baushenko, Artem Snegirev, Tatiana Shavrina, Sergey Markov, Vladislav Mikhailov, and Alena Fenogenova. 2023. A family of pretrained transformer language models for russian.