DeepMistake: Which Senses are Hard to Distinguish for a Word-in-Context Model

Nikolay Arefyev[◇△▽]
Moscow, Russia
nick.arefyev@gmail.com

Maksim Fedoseev[◇]
Moscow, Russia
maxim.fedoseev13@gmail.com

Vitaly Protasov[○]
Moscow, Russia
vitaly.protasov@skoltech.ru

Daniil Homskiy[◇]
Moscow, Russia
homdanil123@gmail.com

Adis Davletov[◇▷]
Moscow, Russia
dev.davletov@gmail.com

Alexander Panchenko[○]
Moscow, Russia
A.Panchenko@skoltech.ru

[◇]Lomonosov Moscow State University
[△]Samsung Research Center Russia
[▽]HSE University
[○]Skolkovo Institute of Science and Technology
[▷]RANEPA

Abstract

In this paper, we describe our solution of the Lexical Semantic Change Detection (LSCD) problem. It is based on a Word-in-Context (WiC) model detecting whether two occurrences of a particular word carry the same meaning. We propose and compare several WiC architectures and training schemes, and also different ways to convert WiC predictions into final word scores estimating the degree of semantic change.

We participated in the RuShiftEval LSCD competition for the Russian language, where our model achieved 2nd best result during the competition. During post-evaluation experiments we improved the WiC model and managed to outperform the best system. An important part of this paper is detailed error analysis where we study the discrepancies between WiC predictions and human annotations and their effect on the LSCD results.

Keywords: semantic change detection, XLM-R, contextualized embeddings
DOI: 10.28995/2075-7182-2021-20-16-30

ГлубокоеНедоразумение: какие значения тяжело различать контекстуализированным моделям значений слов

Арефьев Николай[◇△▽]
Москва, Россия
nick.arefyev@gmail.com

Федосеев Максим[◇]
Москва, Россия
maxim.fedoseev13@gmail.com

Протасов Виталий[○]
Москва, Россия
vitaly.protasov@skoltech.ru

Хомский Даниил[◇]
Москва, Россия
homdanil123@gmail.com

Давлетов Адис[◇▷]
Москва, Россия
dev.davletov@gmail.com

Панченко Александр[○]
Москва, Россия
A.Panchenko@skoltech.ru

[◇]Московский Государственный Университет им. М. В. Ломоносова
[△]Московский Исследовательский Центр Самсунг
[▽]Национальный исследовательский университет «Высшая школа экономики»
[○]Сколковский Институт Науки и Технологий
[▷]РАНХиГС

Аннотация

В этой статье мы описываем наше решение проблемы обнаружения семантических сдвигов значений слов (LSCD). Оно основано на модели Word-in-Context (WiC), которая по двум вхождениям одного слова определяет, имеют ли эти вхождения одно значение. Мы предлагаем и сравниваем несколько архитектур и схем обучения WiC, а также различные способы преобразования предсказаний WiC в итоговые оценки для слов, отражающие степень их семантического сдвига.

Мы участвовали в соревновании RuShiftEval по обнаружению семантических сдвигов значений слов русского языка, где наше решение заняло 2ое место. После завершения соревнования мы улучшили нашу модель WiC и смогли превзойти лучшую систему. Важной частью этой статьи является подробный анализ ошибок, в котором мы изучаем расхождения между прогнозами WiC и аннотациями людей, а также влияние этих расхождений на результаты LSCD.

Ключевые слова: обнаружение семантических изменений, XLM-R, контекстуализированные вектора слов

1 Introduction

The task of Lexical Semantic Change Detection (LSCD) is to determine how senses of a particular word changed between two time periods. The change of word senses in time is a rather complex phenomenon. Thus, several formal settings exist for this task with different annotation schemes and quality metrics, each having its own pros and cons. We developed our system for the RuShiftEval competition [5], where the main goal was to rank the given test words similarly to the ranking by their gold COMPARE scores [11]. To obtain the gold COMPARE scores during the construction of the dataset, the annotators were presented several dozen sentence pairs, each representing two occurrences of the same word, one sampled from an old corpus and another from a new corpus. The annotators estimated the similarity of those word occurrences in meaning on a 1-4 scale, where larger values corresponded to higher degree of similarity. This is commonly known as the Word-in-Context (WiC) task [7]. To obtain gold COMPARE score for a particular word, the annotations of sentence pairs containing this word were averaged.

We decided to follow the same word scoring procedure, but replaced human annotators with a WiC model to solve the task. Our system achieved the second-best result during the competition. After the competition, we managed to outperform the winner by improving the architecture and the training procedure of our WiC model.

Our main contributions are the following.

- An approach to the RuShiftEval LSCD task employing a WiC model is proposed. Our system implementing this approach achieved the 2nd best result in the competition and outperformed the winner after the competition.
- We proposed different architectures and training schemes of a WiC model and compared their performance in the LSCD task.
- A detailed error analysis is performed, showing which distinctions between word senses a WiC model annotates differently compared to human annotators and how it effects the final LSCD results.

2 Related work

RuShiftEval [5] is the first LSCD shared task for the Russian language. Its data annotation scheme and metrics follow those proposed as a part of Diachronic Usage Relatedness (DURel) dataset for German [11]. In [9] the RuSemShift dataset is introduced, which was proposed as the training and the development set for RuShiftEval. In our work, we employed the COMPARE scores from that dataset, which estimate the similarity in meaning between two time periods both for words and sentence pairs. We left out other potentially useful information about the variability of meaning inside each time period. An alternative dataset annotation procedure and metrics were proposed in SemEval-2020 Task 1 [10], where the authors tried to account for the appearance or disappearance of relatively rare word senses, which the COMPARE metric is not sensitive to. They basically clustered word occurrences corresponding to their senses using human annotators instead of an automated clustering system. Based on this gold standard clustering, two subtasks were proposed. The first subtask required binary classification determining if the set of word senses has changed. The second subtask required ranking words according to the change in sense frequencies.

In [9] several LSCD models are compared for the Russian language. One of them is ELMo [6], which is a recurrent neural network trained as a language model on texts from the Russian National Corpus [1] (RNC) and used to build contextualized embeddings of target words. Then they compute the cosine similarity

[1] https://ruscorpora.ru

or the Jensen-Shannon divergence to receive the semantic change score. The best quality among the models considered by them according to the COMPARE metric is 0.403 for the first part of RuSemShift and 0.541 for the second part. Since we have split the RuSemShift dataset into training and development parts, our results are not directly comparable, though the experiments in Section 4 suggest that our system significantly improves those results for RuSemShift.

Next we describe best LSCD methods proposed for the SemEval-2020 Task 1 [10].

UG Student Intern team[10] achieved the 1st place in subtask 2 (ranking). They use word2vec SGNS word embeddings with the Orthogonal Procrustes alignment and compute Euclidean distance instead of cosine distance to evaluate the semantic change. Unfortunately, this team did not publish a system description paper.

Jiaxin & Jinan team [12] achieved the 3rd place in subtask 1 (binary classification) and the 2nd place in subtask 2. They use Temporal Referencing [3] to solve the alignment problem. Instead of training two models and then aligning, they train one model, in which the postfix "_new" is added to the target words from the examples of the new time period, and the postfix "_old" is added to the words from the examples of the old time period. To get a threshold for the classification task, they fit a Gamma distribution by the cosine distance (Gamma Quantile Threshold method). Word embeddings were extracted from fine-tuned BERT or SGNS.

UWB team [8] achieved the 1st place in subtask 1 and the 4th place in subtask 2. They use Canonical Correlation Analysis (CCA) and modification of the Orthogonal Transformation from VecMap for linear transformation to move from the source space (first time period) to the target space (second time period). For word embeddings, they also employ SGNS. After a linear transformation, they calculate the cosine similarity. They proposed different ways to find an optimal threshold based on averaging cosine similarities for each word.

3 Semantic change detection method

To estimate the COMPARE score of a particular target word during the dataset construction, the pairs of sentences were sampled from two time periods. For each pair of sentences, each annotator specified a number from 1 to 4, where 1 stands for unrelated word meanings, and 4 stands for identical meanings. Those annotations were averaged across annotators first, and then across all sentence pairs containing the target word. To approximate this process of computing word scores, our system employs a Word-in-Context model, which solves the same task as the annotators. The input data for this model consists of a target word and two sentences in which the word appears. The model determines whether the target word is used in the same sense or different senses.

First, we have built and trained a WiC model. Then for each test word we retrieved sentences containing this word, and constructed pairs of sentences belonging to different time periods. The scores for sentence pairs were obtained from the WiC model and aggregated into the final word scores.

3.1 Construction of WiC sentence pairs

For each test word, we retrieved the examples from the diachronic subset of the RNC corpus[2] (RNC). This corpus consists of three parts: *Soviet, Pre-Soviet, Post-Soviet*. Since the corpus contains only plain texts, to find examples for a particular word in all forms we used Rulemma lemmatizer[3].

Next, we sampled 100 sentences (or all sentences, if there were fewer) from each time period and constructed sentence pairs for each pair of periods. The examples were sampled from a uniform distribution. For each word, we removed 25% of the longest sentences, 25% of the shortest sentences, and all sentences where the target word was the first or the last word. This was done based on the intuition that for optimal WiC performance, the context shall be long enough, but not very long for faster processing, and there shall exist some preceding and succeeding words for the target word, which often provide the most informative context. In appendix B additional experiments with removal of short and long examples are described.

[2] https://ruscorpora.ru/new/en/corpora-usage.html

[3] https://github.com/Koziev/rulemma

3.2 Scoring sentence pairs

3.2.1 WiC model architecture

As a backbone for our WiC model we employed the XLM-R masked language model [2], which was pre-trained on about 2TB of texts in 100 languages. This enables us using WiC training data in different languages, which improves the overall performance. To score each sentence pair we feed it to XLM-R in the standard format:

```
<s>sentence1</s>sentence2</s>
```

After that we calculate the contextualized embeddings for the target word in each sentence by averaging the outputs of the last transformer layer corresponding to all of its subwords (*mean* pooling). Additionally we experimented with the $first$ pooling when the output on the first subword is taken.

Then we aggregate two target word embeddings from two sentences using one of the following options:

1. **concat**: (x, y), the concatenation of two embeddings;
2. **comb_dmn**: $(x-y, \bar{x} \circ \bar{y})$, the concatenation of the difference (**d**) of non-normalized and component-wise product (**m**) of normalized (**n**) embeddings;
3. **dist_l1**: $\|x - y\|_1$, L1-distance between the embeddings, also known as the Manhattan distance;
4. **dist_l1ndotn**: $(\|\bar{x} - \bar{y}\|_1, \langle \bar{x}, \bar{y} \rangle)$, the concatenation of L1-distance and the dot product (**dot**) of the normalized (**n**) embeddings. The second feature is essentially the cosine similarity.

While the first two options produce high-dimensional vectors, the other two result in a vector with one or two components only. The obtained vector is passed through a classification head, which has a dense layer of size hs and $tanh$ activation, followed by a linear layer, or only a linear layer (denoted as $hs = 0$). For the first two aggregation options, we always used a hidden layer of the size $hs = 1024$ (the size of the embeddings in large XLM-R). For the distance-based inputs we found that a linear head outperformed non-linear one. We inserted batch normalization [4] before the first layer, which proved to be especially efficient for L1-distance inputs.

Based on the intuition that the similarity between two-word occurrences does not depend on the order of sentences in a sentence pair, we employed training time and test time augmentation. For each example, we create an additional one by swapping two sentences. Thus, two scores were obtained for each example. For training, we always use that augmentation since, from our preliminary experiments, it helps for some architectures and never hurts. For inference we either take the first score (i.e. disable test time augmentation), or average them.

3.2.2 WiC training

We also look at different ways to train the model using MCL-WiC[4] and RuSemShift [9] datasets. **MCL-WiC** consists of an English training set (8000 examples), multilingual development sets with both sentences in one of the following languages: English, French, Russian, Arabic, Chinese (1000 ex. each), and test sets with cross-lingual (one sentence in English, the second in another language) and multilingual parts (1000 ex. each). **RuSemShift** consists of two pairs of periods: there are pairs of sentences from pre-Soviet and Soviet periods in the first part, and Soviet and post-Soviet in the second part. We have split each part into a train and a development subsets, ensuring there is no intersection between the target words in those subsets (lexical split).

The weights of the pre-trained XLM-R large model were used for initialization, and then we train model on the following datasets or their combinations.

1. **MCL-WiC** training set consists of the original MCL-WiC training set in English, 70% of each non-English development set (2800 ex.) and all trial sets (72 ex.). The development set is the rest 30% of non-English development sets (1200 ex.) and the full English development set (1000 ex.).
2. **MCL-WiC en-en** training set is the original MCL-WiC training set. It is used to estimate the performance of a model trained only on English WiC data.
3. **MCL-WiC ru-ru** means that we train only on data in Russian, including 70% of the development set and the whole test and trial sets (1708 ex. in total).

[4] https://github.com/SapienzaNLP/mcl-wic

4. **RuSemShift** training set is combined from both training sets from our split, 3898 examples in total. The development set consists of two parts - one for pre-Soviet and Soviet periods, another for Soviet and post-Soviet periods.

The training process consists of one or two steps. Each step employs its own training set and loss function. All training schemes are enumerated in Table 1. Cross entropy and mean squared error losses are denoted as CE and MSE accordingly.

To employ all training data we have in Russian simultaneously, we have developed a new loss function $MSE+$, which can handle both binary targets from MCL-WiC and real-valued targets from RuSemShift. For the real-valued targets, it is equivalent to MSE loss, while for binary targets, it penalizes the predictions using MSE loss only when they are outside (1,2) interval for negative examples or outside (3,4) interval for positive. Since the labels are binary, we only know whether the meaning is similar or different, but do not know the exact degree of similarity, thus any prediction from the appropriate intervals is suitable. Another option is binarizing RuSemShift and using CE loss. Examples with scores not less than 3 were treated as positive. For negative examples we set the threshold of 2 during the competition and 3 in the following experiments.

Train#1	Loss#1	Train#2	Loss#2
MCL-WiC	CE	-	-
MCL-WiC en-en	CE	-	-
MCL-WiC ru-ru	CE	-	-
RuSemShift	MSE	-	-
MCL-WiC	CE	MCL-WiC ru-ru	CE
MCL-WiC	CE	RuSemShift	MSE or CE
MCL-WiC	CE	RuSemShift + MCL-WiC ru-ru	MSE+ or CE

Table 1: Training schemes. We take XLM-R pre-trained as a MLM, fine-tune it first on Train#1 with Loss#1, and then optionally on Train#2 with Loss#2.

3.3 Scoring words

Mean. The simplest method to compute the final score for a particular word and a pair of periods is to calculate the mean of scores for all corresponding sentence pairs. across all pairs of sentences containing the target word for a pair of periods.

Isotonic regression (IsoReg). Depending on the loss function, the predicted scores for sentence pairs may not be in the same range or distribution as the human scores. This may result in incorrect word raking after simple averaging. We try to make sentence pair scores more similar to human scores by fitting isotonic regression [1]. We feed the predicted score for a sentence pair to the isotonic regression and get the modified score, which is then averaged across sentence pairs. Isotonic regression is trained on sentence pairs from the training subset of RuSemShift (3989 training examples).

Linear regression (LinReg). Instead of using simple averaging of scores for sentence pairs, we can use a trainable function to predict word scores. Input features are the mean and quartiles of scores for all sentence pairs of a particular word and pair of periods. We trained this model on words from the training subset of RuSemShift (69 training words). The features can be calculated both on sentence pairs from RuSemShift, or sampled sentence pairs (**LinReg_s**).

4 Experiments and results

To evaluate our WiC model and select its hyperparameters, we employed two types of metrics. Spearman correlation between model scores and gold scores for sentence pairs from RuSemShift (***sentSpear***) shows how well the model solves WiC task. Spearman correlation between the final word scores and the gold values of the COMPARE metric (***wordSpear***) estimates the final performance on LSCD task.

Both metrics are calculated on each of two development sets (***dev1*** for pre-Soviet – Soviet, ***dev2*** for Soviet – post-Soviet) and three test sets (***p12*** for pre-Soviet – Soviet, ***p23*** for Soviet – post-Soviet, ***p13*** for pre-Soviet – post-Soviet). For majority of experiments we show averaged dev and a test metrics. In the experiments with WiC model architecture and training, for evaluation we employed the same sentence pairs that were annotated by humans, otherwise we could not calculate *sentSpear*. However, for the final results in Table 2 and word scoring experiments in Section 4.3, 100 sampled pairs for each word and pair of periods were used instead. Additionally, when training on MCL-WiC we employ accuracy on the English dev set (***en-acc***), or an average accuracy over non-English dev sets (***nen-acc***) for early stopping.

Method/Team	**Avg**	**p12**	**p23**	**p13**
Best results of other teams				
GlossReader (1st best result)	**0.802**	0.781	**0.803**	**0.822**
vanyatko (3rd best result)	0.720	0.678	0.746	0.737
Our submissions: team *DeepMistake* (2nd best result)				
first+concat on MCL_{CE}^{en-acc} →$\text{RSS}_{MSE}^{dev2-sentSpear}$ (***M1***), LinReg	**0.791**	**0.798**	0.773	0.803
M1, Mean	0.789	0.794	0.773	0.799
M1, IsoReg	0.789	0.793	0.775	0.798
p12, p13: *M2*; *p23*: first+concat on $\text{MCL}_{CE}^{nen-acc}$ →RSS+ruMCL$_{CE}^{dev1-sentSpear}$, IsoReg	0.785	0.773	**0.802**	0.780
mean+dist_l1ndotn-hs300 on $\text{MCL}_{CE}^{nen-acc}$ → RSS+ruMCL$_{MSE+}^{dev1-sentSpear}$ (***M2***), Mean	0.780	0.773	0.786	0.780
LinReg on *M1* + *M2* + *M3*	0.780	0.756	0.772	**0.811**
p12, p13: mean+dist_l1 on $\text{MCL}_{CE}^{nen-acc}$ →$\text{RSS}_{MSE}^{dev2-sentSpear}$ (***M3***), Mean *p23*: first+concat on $\text{MCL}_{CE}^{nen-acc}$ →RSS+ruMCL$_{CE}^{dev2-sentSpear}$, Mean	0.779	0.749	0.801	0.788
p12, p13: *M2*; *p23*: max+concat on $\text{MCL}_{CE}^{nen-acc}$ →$\text{RSS}_{MSE}^{dev2-sentSpear}$, Mean	0.778	0.779	0.775	0.779
p12, p13: *M3*, LinReg* *p23*: mean+comb_dmn on MCL_{CE}^{en-acc} →$\text{RSS}_{MSE}^{dev2-sentSpear}$, LinReg	0.757	0.750	0.732	0.788
Our best models with ablation analysis				
mean+dist_l1ndotn-hs0 on $\text{MCL}_{CE}^{nen-acc}$ →$\text{RSS}_{MSE}^{dev2-sentSpear}$, Mean	**0.823**	**0.825**	**0.821**	**0.823**
mean+dist_l1ndotn-hs0 on $\text{MCL}_{CE}^{nen-acc}$ →RSS+ruMCL$_{MSE+}^{dev2-sentSpear}$, Mean	0.803	0.800	0.798	0.811
mean+dist_l1ndotn-hs0 on $\text{MCL}_{CE}^{nen-acc}$, Mean	0.776	0.777	0.778	0.772
mean+concat on $\text{MCL}_{CE}^{nen-acc}$ →$\text{RSS}_{MSE}^{dev2-sentSpear}$, Mean	0.768	0.760	0.759	0.784
mean+concat on $\text{MCL}_{CE}^{nen-acc}$ →RSS+ruMCL$_{MSE+}^{dev2-sentSpear}$, Mean	0.791	0.790	0.786	0.797

Table 2: Evaluation and post-evaluation results. MCL is MCL-WiC, RSS is RuSemShift. *LinReg** denotes LinReg on two features only - the mean and the median. M1, M2, M3 abbreviate duplicated WiC model specifications. *LinReg on M1+M2+M3* denotes LinReg on features from all those models. **concat**: (x, y), **comb_dmn**: $(x - y, \bar{x} \circ \bar{y})$, **dist_l1**: $\|x - y\|_1$, **dist_l1ndotn**: $(\|\bar{x} - \bar{y}\|_1, \langle \bar{x}, \bar{y} \rangle)$

4.1 Submissions and post-competition improvements

During the evaluation phase, we made 10 submissions. Their results with the best results of other teams are shown in Table 2. The first 2 arguments of the method name indicate which pooling and aggregation of the two target word embeddings were used. Then the training scheme is specified with subscript and superscript specifying loss function and early stopping metric. The word scoring method is appended after the comma when it differs from the default mean over sentence pairs. In some submissions, for different pairs of time periods we used predictions of different models.

After the competition, we analyzed various aggregation methods of the target word embeddings and training options and managed to achieve better results than the winning submission for all pairs of time periods. The best model uses *dist_l1ndotn* without hidden layer (*hs0*) and is trained on MCL-WiC first, then on RuSemShift with MSE loss. From ablations we notice that the average quality is reduced by 5 points when only the first training step is left. For *dist_l1ndotn* is better to train on RuSemShift with MSE loss than on RuSemShift+ruMCL-WiC with $MSE+$ loss, but for *concat* vice versa.

In appendix A we additionally compare the performance of our best model with human performance.

4.2 WiC architecture and training scheme

Embeddings aggregation. To compare different methods of aggregation of target word embeddings, we trained several WiC models on MCL-WiC, and then optionally fine-tuned them on RuSemShift with MSE loss. For early stopping we employed *nen-acc* on the first dataset and *dev2-sentSpear* on the second. We used mean pooling for subwords and also Mean aggregation of scores for sentence pairs.

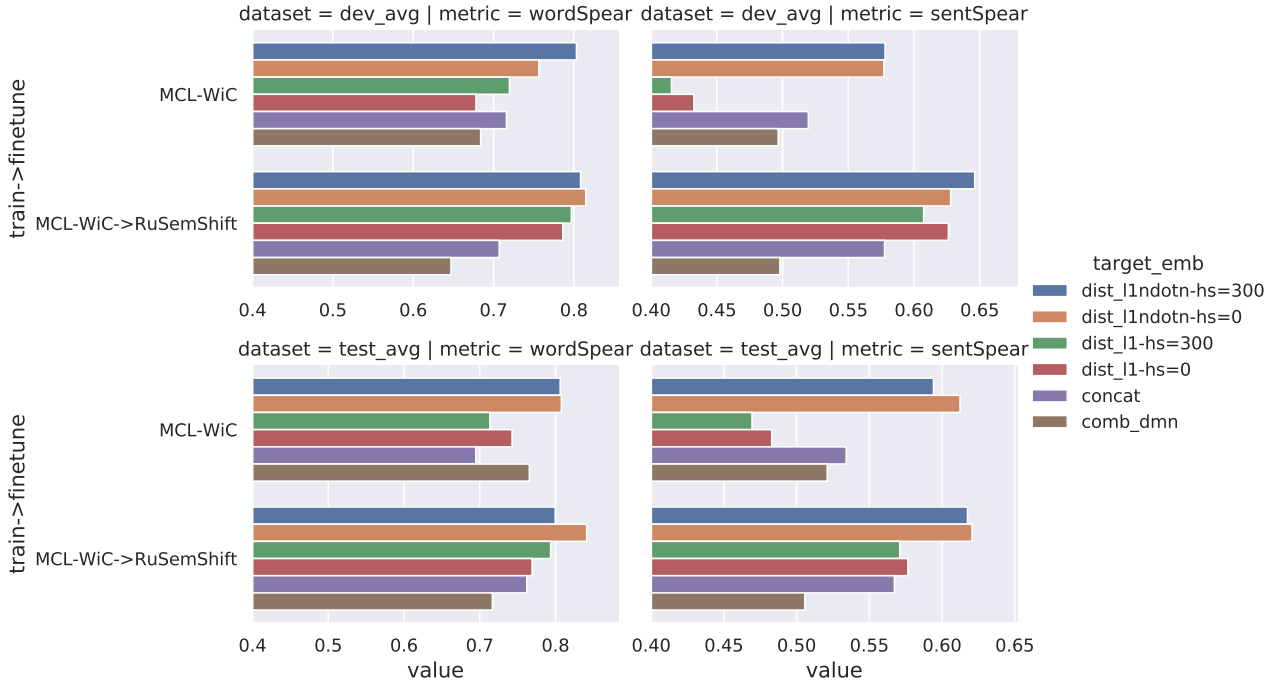

Figure 1: Comparison of the target embedding aggregations. Values are sentSpear (left) and wordSpear (right) on dev (up) and test (down).

Figure 1 shows that the best methods of combining two embeddings of the target word are $dist\_l1ndotn$ either with dense layer size $hs = 300$ or without dense layer. Generally, training on RuSemShift after training on MCL-WiC improves performance a bit or at least does not hurt. And for the two-step training, the basic one-dimensional $dist\_l1$ works better than high dimensional concatenation or $comb\_dmn$. Moreover, concatenation always works better than $comb\_dmn$ for two-step training, but when training only on MCL-WiC $comb\_dmn$ gives higher *test-wordSpear*.

WiC model training. Figure 2 compares different training schemes. All compared models employ the $mean$ subword pooling, the concatenation of target word embeddings and *dev2-sentSpear* for early stopping. Training schemes are specified in the following format.

- In the case of one-step training: ”training dataset” (loss function of the training).
- In the case of two-step training: ”training dataset #1” -> ”training dataset #2” (loss function of the second training). The loss function of the first step is *CE* by default.

Evidently, two-step training procedure employing both the large multilingual MCL-WiC dataset and the task-specific RuSemShift dataset significantly boosts the performance compared to single-step training on any of the datasets alone. Using the combination of RuSemShift and the part of MCL-WiC in Russian with the proposed *MSE+* loss for the second training step generally gives the best overall performance, except for the *dev-wordSpear*, which is a little better for another scheme presumably due to the metric variance. Using RuSemShift on the second training step shows much better performance than employing the Russian part of MCL-WiC for the same purpose. For the single-step training schemes, we observe a large difference between dev and test performance for model trained on only English or Russian parts of MCL-WiC. The model trained on RuSemShift with MSE loss ranks sentences much worse than the one trained on MCL-WiC, but their results of the final word ranking are comparable. Surprisingly,

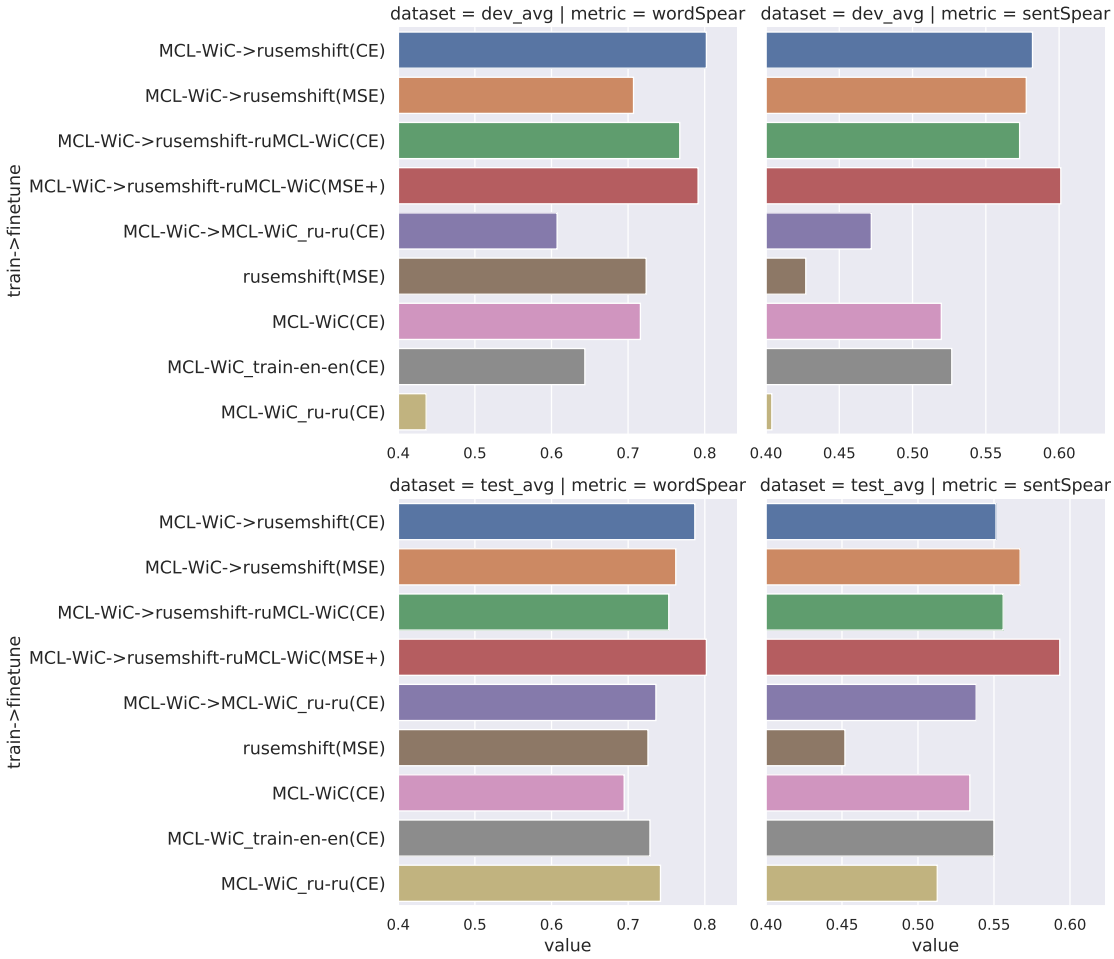

Figure 2: Comparison of the WiC model training. Aggregation of the target word embeddings is *concatenation*.

one can train the model on the English part of MCL-WiC only and still obtain a relatively good LSCD results comparable to the 3rd best team in the competition. This shows strong zero-shot cross-lingual transfer capabilities of the underlying XLM-R model. Training on the Russian part of MCL-WiC alone gives mixed results presumably due to much smaller size of this part.

4.3 WiC scores aggregation for word scoring

Figure 3 shows the comparison of different methods of obtaining word scores from sentence pairs scores. The quality does not depend on the method of word scoring as much, as on the WiC model architecture or training scheme, but the linear regression gives consistently the best or nearly the best results.

Finally, we estimated how the quality of the final word ranking depends on the number of sentence pairs sampled for each word. We sampled each number of sentence pairs 30 times and calculated the mean and the standard deviation of the target *wordSpear* metric. Figure 4 shows the results for test words. As we expected, the target metric improves rapidly with the number of sampled sentences, and also its standard deviation decreases. Even for 80 samples std is 0.8 point, suggesting that different sampled pairs result in 2-3 point difference in the target metrics. This figure also suggests that if only several dozen of sentence pairs were annotated for each word during the construction of a LSCD dataset, the difference between methods of 5-10 points may be due to chance. The green dashed line shows the results when we run our system on the same sentence pairs that were annotated by humans. Unsurprisingly, this results in better

estimate of the gold word scores. However, the difference becomes small as the number of sampled pairs approaches one hundred.

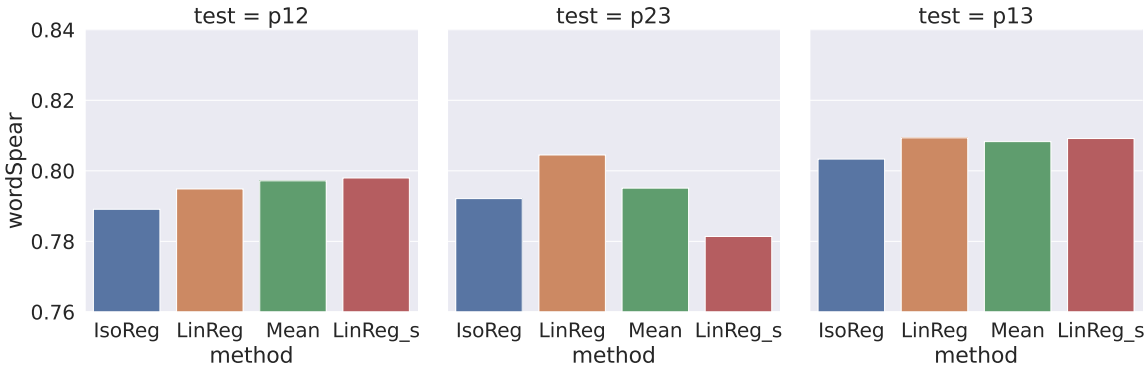

Figure 3: Word scoring methods. Results for the predictions on the sampled test pairs. Model: dist_l1ndotn-hs0 on $\text{MCL}_{CE}^{nen-acc}$ →RSS+ruMCL$_{MSE+}^{dev2-sentSpear}$

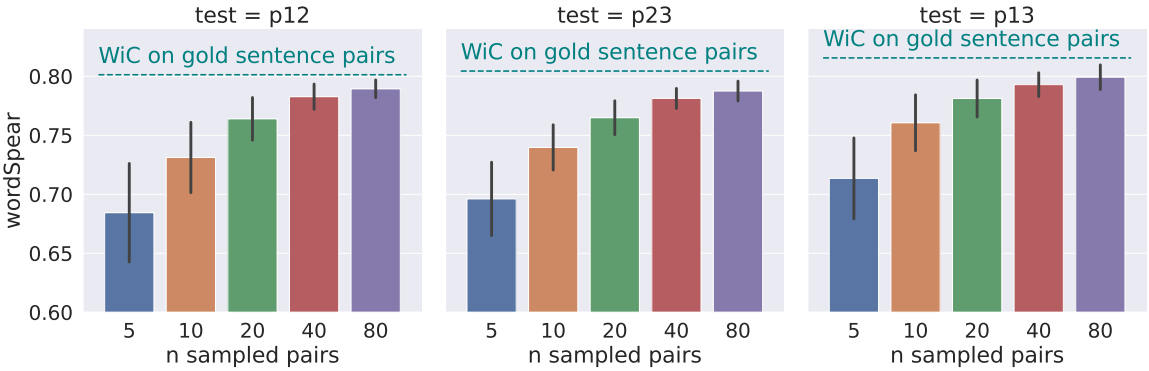

Figure 4: Dependence on the number of sampled pairs. Error bars show one standard deviation. Model: mean+dist_l1ndotn-hs0 on $\text{MCL}_{CE}^{nen-acc}$ →RSS+ruMCL$_{MSE+}^{dev2-sentSpear}$

5 Error analysis

This section is devoted to getting some insights into the types of errors, their relative frequencies and reasons. We used the test set consisting of 99 unique target words and sentence pairs for them with human annotations, which was provided by the organizers after the competition. We employed the WiC model with first subword pooling and concatenation of target embeddings trained on $\text{MCL}_{CE}^{nen-acc}$ →RSS+ruMCL$_{CE}^{dev1-sentSpear}$ with *mean* word scoring. This model achieved one of the best results among our submissions.

We define $\Delta Rank$ as the difference between the rank of a word predicted by our model and the gold rank. Words with a high $|\Delta Rank|$ value are considered serious errors, we decided to focus on the words with $|\Delta Rank| \geq 25$. This resulted in 24 words from *p12*, 18 words from *p23*, and 13 words from *p13*. Many of those words are incorrectly ranked in several pairs of time periods, thus, there are 27 unique words that we analyzed in total. They are shown in Figure 5.

5.1 Classification of WiC model mistakes

To understand the reasons of incorrect ranking of words under consideration, for each of them we have selected 5-7 annotated sentence pairs with the highest difference between gold annotations and the predicted WiC scores (the difference was 1.5 at least). We obtained 171 pairs of sentences in total, and annotated them according to the error types described below.

Figure 5: Words with large difference between the predicted and the gold ranks at least for one pair of epochs.

Table 3 shows the results of error analysis and examples. More examples can be found in appendix C. We identified four typical reasons (error types) of the high disagreement between WiC model predictions and human annotations.

1. *Model can not find the difference*. The model incorrectly classifies two word occurrences as having the same meaning.
2. *Model sees wrong difference*. The model incorrectly classifies two word occurrences as having different meanings.
3. *Model seems to be right*. These are pairs of sentences, that in our opinion were correctly classified by the model, but incorrectly annotated by one or more annotators.
4. *Ambiguity*. From the context we could not understand whether two word occurrences have the same meaning.

The most frequent error type (39% of all analyzed examples) is *Model can not find the difference*. This strongly effects the final ranking of words like *тачка* (car / cart), *увольнение* (dismissal / vacation), *дядька* (servant tutor / uncle / mister) that obtained or lost the first sense, which the model cannot distinguish from others.

Error types *Model seems to be right* and *Model sees wrong difference* have almost equal frequency (23.2% and 22.8%). In the sentences of the first type, the target word usually has two senses that are sim-

Table 3: Types of errors, their proportions and examples.

Type	Gold	Model	Pair of sentences
Model can not find the difference. **39%**	1 1 1	4	Пазульский был приговорен в Одессе к 12 годам, 100 плетям и трем годам прикования к **тачке**. Они с Верой выпорхнули из дверей **тачки** и поплыли в невесомости спортивного зала.
Model seems to be right. **23.2%**	1 3 3	1	”Если гомеопаты не обходятся без прививок, то чего же нам стесняться!” – так утешают себя **маньяки** прививок ... Только преследований **маньяка** мне в таком состоянии не хватало
Model sees wrong difference. **22.8%**	4 4 4	1	На руках она с усилием тащила Федьку, прижав его поперек **живота**, чем он нисколько не смущался. Боли в **животе** не то стали слабее, не то он к ним привык
Ambiguity. **15%**	1 4 3	4	Призыв 1902 года в 1924 году дал Красной армии 4.700 партийцев; при **увольнении** же этого возраста в 1926 году в запас Красная армия дала стране 19.439 партийцев. **Увольнение** производится в порядке очередности.

ilar to some degree. Often there is disagreement between annotators in such cases. Since there were only three annotators, even one incorrect annotation significantly effected the resulting mean human score for a sentence pair. For instance, Table 3 contains sentences for the word *маньяк* (maniac), which has a direct meaning (a mentally ill man) and a figurative meaning (a person obsessed with a passionate attraction to something). The model correctly distinguished these senses, but two out of three annotators decided that those senses are very similar. For sentence pairs of the type *Ambiguity* there is large disagreement between annotators, hence, the aggregated gold annotation is almost random. For instance, in Table 3 the word *увольнение* (dismissal / vacation) in the second sentence can express any of its meanings.

This error analysis suggests that about 60% of the analyzed sentences are actually incorrect predictions, while the rest are hard cases where human annotators disagree with each other. Such cases can be easily found by high deviation between annotations, and it may be beneficial involving additional annotators to resolve disagreement or to filter unclear examples.

Another technical issue we found were incorrectly tagged examples in the test set. The organizers of the competition published the test set with annotated sentences. It consists of three files, each of them contains approximately 3000 pairs of sentences containing some target word highlighted by special tags **, **. However, in a significant proportion of sentences (Table 4) the target word was not tagged, which was a problem for our WiC model. Finally, we fixed the largest part of incorrect examples, the rest consisted of sentences with abbreviated target words, for example: *апостол* $\rightarrow$ *ап.*, *век* $\rightarrow$ *в.* Our fixes are merged into the published version of the dataset, hopefully, making the dataset better.

Table 4: Incorrectly tagged examples.

Test set	Total examples	Bad examples	Fixed examples
p12	2965	236	214
p23	2967	221	191
p13	2969	249	207

6 Conclusion

We have proposed an approach to Semantic Change Detection employing a Word-in-Context model and found that it has strong performance achieving the 2nd best result among other competing approaches, which can be further improved to outperform the 1st best result by improving WiC architecture and training procedure. Regarding the architecture, we have found that a simple linear head on top of concatenated L1 distance and dot product between contextualized XLM-R embeddings provides better performance than more common alternatives like embedding concatenation and non-linear classification heads. For

the training procedure, using training on a large multilingual WiC dataset first and fine-tuning on a task-specific RuSemShift data later results in the best overall performance.

We performed a detailed error analysis of sentences, where disagreement between model and annotators was the highest. To understand why the model fails, we annotated 171 pairs of sentences and revealed four different types of low model results. Such mistakes included examples when the model correctly distinguished the difference or similarity of senses, when annotators were wrong, and vice versa.

References

[1] Miriam Ayer, H. D. Brunk, G. M. Ewing, W. T. Reid, and Edward Silverman. An Empirical Distribution Function for Sampling with Incomplete Information. *The Annals of Mathematical Statistics*, 26(4):641 – 647, 1955.

[2] Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. Unsupervised cross-lingual representation learning at scale. *arXiv preprint arXiv:1911.02116*, 2019.

[3] Haim Dubossarsky, Simon Hengchen, Nina Tahmasebi, and Dominik Schlechtweg. Time-out: Temporal referencing for robust modeling of lexical semantic change. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 457–470, Florence, Italy, July 2019. Association for Computational Linguistics.

[4] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift, 2015.

[5] Andrey Kutuzov and Lidia Pivovarova. Rushifteval: a shared task on semantic shift detection for russian. *Komp'yuternaya Lingvistika i Intellektual'nye Tekhnologii: Dialog conference*, 2021.

[6] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations, 2018.

[7] Mohammad Taher Pilehvar and Jose Camacho-Collados. WiC: the word-in-context dataset for evaluating context-sensitive meaning representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1267–1273, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[8] Ondřej Pražák, Pavel Přibáň, Stephen Taylor, and Jakub Sido. UWB at SemEval-2020 task 1: Lexical semantic change detection. In *Proceedings of the Fourteenth Workshop on Semantic Evaluation*, pages 246–254, Barcelona (online), December 2020. International Committee for Computational Linguistics.

[9] Julia Rodina and Andrey Kutuzov. RuSemShift: a dataset of historical lexical semantic change in Russian. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 1037–1047, Barcelona, Spain (Online), December 2020. International Committee on Computational Linguistics.

[10] Dominik Schlechtweg, Barbara McGillivray, Simon Hengchen, Haim Dubossarsky, and Nina Tahmasebi. SemEval-2020 task 1: Unsupervised lexical semantic change detection. In *Proceedings of the Fourteenth Workshop on Semantic Evaluation*, pages 1–23, Barcelona (online), December 2020. International Committee for Computational Linguistics.

[11] Dominik Schlechtweg, Sabine Schulte im Walde, and Stefanie Eckmann. Diachronic usage relatedness (DURel): A framework for the annotation of lexical semantic change. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 169–174, New Orleans, Louisiana, June 2018. Association for Computational Linguistics.

[12] Jinan Zhou and Jiaxin Li. TemporalTeller at SemEval-2020 task 1: Unsupervised lexical semantic change detection with temporal referencing. In *Proceedings of the Fourteenth Workshop on Semantic Evaluation,* pages 222–231, Barcelona (online), December 2020. International Committee for Computational Linguistics.

A Comparison with human quality

It is interesting to compare the performance of our best model with human performance. There are three columns containing annotations of each sentence pair by different annotators in RuShiftEval [5] and five columns in RuSemShift [9]. To estimate human performance, we excluded examples with one or more annotations absent or having zero (undecided) values. For examples with full set of annotations we calculated *wordSpear* and *sentSpear* between each column and the mean of other columns (*annotN VS mean_wo_N*). However, this method of human performance estimation shall be taken with a grain of salt. It is totally correct only if each human annotated the whole dataset, which may not be true for RuSemShift and RuShiftEval datasets that were annotated using crowdsourcing. We suppose, that each of three or five annotators specified in the datasets really correspond to several humans. In this case we are probably overestimating human performance of word ranking (*wordSpear*), because after averaging across all sentence pairs individual misconceptions about senses of a particular word resulting in incorrect annotations of some sentence pairs with this word by one human will be partially compensated by annotations of other sentence pairs with the same word obtained from other humans. To get better estimates of human performance for datasets annotated with crowdsourcing, additional identifiers of humans who annotated each example are required, which are not available for these two datasets. However, we still believe that our estimates may be useful to some degree, even if they are just an upper bound. For comparison in table 7 we provide estimates of human performance obtained by the same procedure for the DURel [11] LSCD dataset in German, which has a structure similar to RuShiftEval and RuSemShift. This dataset was fully annotated by each of five annotators, thus, our procedure shall estimate human performance correctly for DURel.

method	wordSpear		sentSpear	
	dev1	dev2	dev1	dev2
annot1 VS mean_wo_1	0.941	**0.964**	0.516	0.587
annot2 VS mean_wo_2	0.940	0.940	0.527	0.545
annot3 VS mean_wo_3	0.943	0.951	0.547	0.545
annot4 VS mean_wo_4	0.939	0.941	0.545	0.558
annot5 VS mean_wo_5	**0.961**	0.923	0.524	0.566
mean+dist_l1ndotn-hs0 on $\text{MCL}_{CE}^{nen-acc} \rightarrow RSS_{MSE}^{dev2-sentSpear}$	0.819	0.811	**0.599**	**0.657**

Table 5: Comparison of our best model with human quality on RuSemShift development set.

method	wordSpear			sentSpear		
	p12	p23	p13	p12	p23	p13
annot1 VS mean_wo_1	0.932	0.953	**0.959**	0.579	0.621	0.628
annot2 VS mean_wo_2	0.920	0.957	0.948	0.578	0.616	0.605
annot3 VS mean_wo_3	**0.936**	**0.958**	**0.959**	**0.597**	0.626	0.616
mean+dist_l1ndotn-hs0 on $\text{MCL}_{CE}^{nen-acc} \rightarrow RSS_{MSE}^{dev2-sentSpear}$	0.825	0.821	0.823	0.596	**0.634**	**0.631**

Table 6: Comparison of our best model with human quality on RuShiftEval.

method	wordSpear	sentSpear
annot1 VS mean_wo_1	0.875	0.670
annot2 VS mean_wo_2	0.883	0.698
annot3 VS mean_wo_3	0.883	0.678
annot4 VS mean_wo_4	0.939	0.745
annot5 VS mean_wo_5	0.943	0.717

Table 7: Human quality on DURel.

Tables 5 and 6 show that our best model performs ranking of sentence pairs similarly or better than humans. However, word ranking results are significantly worse than those of humans. We suppose that such discrepancy may be due to different biases in mistakes made by humans and our model while scoring sentence pairs. Probably, humans underestimate and overestimate similarity between word occurrences with similar probabilities, so their mistakes are less biased and cancel each other when the average of scores for sentence pairs is calculated to produce word scores. In contrast, as we observed from the error analysis our model in principal does not see differences between some senses. Thus, it consistently overestimates similarity between occurrences of those senses and averaging does not help.

Comparing our estimates of human performance for the Russian datasets and DURel, we observe that humans have better agreement in ranking of sentence pairs (*sentSpear*) on DURel, especially annotators 4 and 5, who were student with a background in historical linguistics. However, our estimates of human performance for word ranking (*wordSpear*) on the Russian datasets is similar or higher than the performance of those two best annotators of DURel. This inderectly supports our hypothesis about overestimation of human performance on the Russian datasets. However, more information about annotators from the crowdsourcing platform is required to draw reliable conclusions.

B Removing short and long sentences

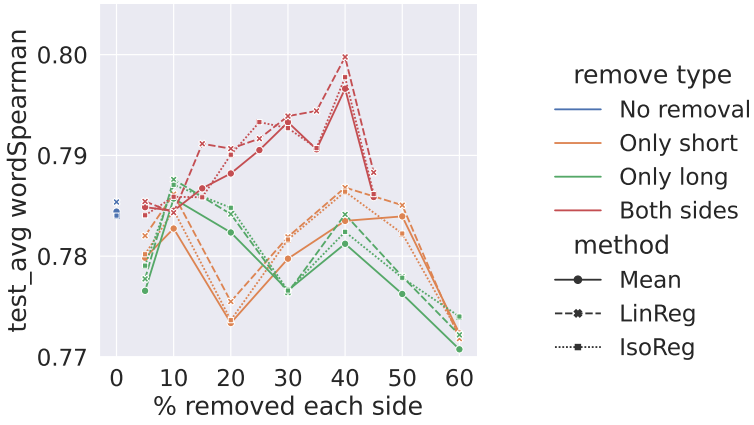

Figure 6: Dependency on percent of removed short and long sentences. Model: first+concat on $\text{MCL}_{CE}^{en-acc} \rightarrow \text{RSS}_{MSE}^{dev2-sentSpear}$

During the competition for each target word we removed 25% of the longest sentences and 25% of the shortest sentences before sampling examples for this word. This was based on our intuition that the WiC model can perform worse on very short or long examples, while thresholds were set arbitrarily. After the competition we decided to study whether removing only shortest or only longest examples is better, and also selecting better threshold. Figure 6 shows the dependence of the competition metric (*wordSpear* on RuShiftEval averaged over three pairs of time periods) on the percent of longest or shortest sentences removed. For symmetric removal the specified percent is removed from each side, resulting in two times more sentences removed. Surprisingly, we observe that removing only shortest or only longest sentences for each word help little. Removing both shortest and longest sentences consistently improve results. The best results are obtained when 40% of the shortest and 40% of the longest sentences are removed for each word, and examples are sampled from only 20% of sentences of medium length.

C More examples of mistakes

Table 8: Samples for mistakes *Model seems to be right* and *Model can not find the difference*

Type	Gold	Model	Pairs of sentences
Model seems to be right. **23.2%**	4 4 3 1 2 2 2 3 4	1 4 1	1) Он, как видно из его стихотворений, взялся за дело поэта по призванию; он сильно сочувствует вопросам своего времени, страдает всеми недугами **века**, болезненно мучится несовершенствами общества и сгорает нетщетно жаждою споспешествовать его совершенствованию и торжеству на земле истины, любви и братства. Во второй половине прошлого **века** рытье колодцев было заменено бурением скважин. 2) Унять было невозможно, по крайней мере в ту минуту, и – вдруг окончательная катастрофа как бомба разразилась над собранием и треснула среди его: третий чтец, тот **маньяк**, который все махал кулаком за кулисами, вдруг выбежал на сцену. Не надо было быть балетным **маньяком**, чтобы понять, что балериной эта особа никогда не будет. 3) Это жертва не моя, а всего **уклада** жизни! Там, верно, рукомойники в сенях, пахнет кухней, мокрые дрова возле печек – убогий, нелюбимый мною, дачный зимний **уклад**.
Model can not find the difference. **39%**	2 1 2 1 1 1 1 2 1 1 1 1 1 4 3	4 4 4 4 4	1) Скоро прибыли к нему **братья** его, Андрей и Борис, с их многочисленною дружиною: не было ни упреков, ни извинений, ни условий; единокровные обнялися с видом искренней любви, чтобы вместе служить отечеству и христианству. Он говорил ей: ”Ты, **брат**”. 2) Злополучный иеромонах был вытащен из **огня** со слабыми признаками жизни. Ночью немцы обрушили на наше расположение массированный артиллерийский **огонь**. 3) В годовщину свадьбы буду выставлять на балконе огненные **цифры**. Сергей Глазьев с **цифрами** в руках наглядно доказал, что идет хищническая добыча нефти и газа, главная цель которой – сверхприбыль любой ценой. 4) Вадима Петровича начинало брать раздражение и на бывшего своего **дядьку**. – Что за ярмарка – Так День Нэзалэжности! – отвечает **дядька** в национальном гуцульском костюме, с приклеенными усами. 5) Команда, за исключением вахтенных, ушла в **увольнение**, в город. Первым плохим признаком стал запущенный в прессу слух, что сразу же после **увольнения** Примакова Рапота сам подал в отставку, а на его место уже подбирается новая кандидатура.

Table 9: Samples of mistakes *Model sees wrong difference* and *Ambiguity*

Type	Gold	Model	Pairs of sentences
Model sees wrong difference. **22.8%**	4 4 4 4 4 4 4 3 4	1 1 1	1) Тут в **кармане** тысяча рублев положена. Вместо птиц он приносил домой целые **карманы** камней и сваливал их под навесом в ящик. 2) Когда Их Величества повернули к той стороне **стены**, которая ведет к Спасским воротам, на площади уже стояла тысячная толпа, приветствовавшая Царя и Царицу восторженными кликами ”ура” и бросаньем в воздух шапок. В конце этого двора у **стены** поставлены бочки, на которые наложены доски. 3) Однажды Петр застал сына в сарае, мальчик пытался пристроить к старому корыту колесо **тачки**. Таня и Освальд не решались сойти с тропы, чтобы не наступить на змею, не потревожить эльфа с крошечной **тачкой**, не поднять из логова сказочного волка.
Ambiguity. **15%**	4 3 4 4 1 4 2 3 1	1 4 4	1) Итак, чтобы покончить с этим предметом, прошу Вас, друг мой, никогда не опасаться задеть мою авторскую **амбицию**. Разговор, конечно, бессмысленный и бесполезный, но наведший все на те же размышления: о Церкви, о Православии, о той мелочной и даже злобной каше интриг, самолюбий, **амбиций**, эгоцентризмов, в которой приходится в Церкви жить. 2) Ребята, с **тачками** туда!.. Скотник, насквозь напитанный запахом навозной жижи, тот самый, что не довез своей **тачки**, а пошел смеяться с мужиками, когда мужики еще добродушно покуривали на крыльцах людской – вертел в руках терракотовую копию химеры: – Ванька, гляди-ка! 3) Понемногу мы привыкали к своему домику и своему новому **укладу**. В странах Европы всегда происходило соприкосновение, а нередко и противоборство, с местной традицией, куда входили и эстетические представления заказчиков, и бытовой **уклад**, и местные ремесленные корпорации.