Studying Attention Models in Sentiment Attitude Extraction Task

Nicolay Rusnachenko[1(✉)] and Natalia Loukachevitch[1,2(✉)]

[1] Bauman Moscow State Technical University, Moscow, Russia
kolyarus@yandex.ru
[2] Lomonosov Moscow State University, Moscow, Russia
louk_nat@mail.ru

Abstract. In the sentiment attitude extraction task, the aim is to identify «attitudes» – sentiment relations between entities mentioned in text. In this paper, we provide a study on attention-based context encoders in the sentiment attitude extraction task. For this task, we adapt attentive context encoders of two types: (I) feature-based; (II) self-based. Our experiments (https://github.com/nicolay-r/attitude-extraction-with-attention) with a corpus of Russian analytical texts RuSentRel illustrate that the models trained with attentive encoders outperform ones that were trained without them and achieve 1.5–5.9% increase by $F1$. We also provide the analysis of attention weight distributions in dependence on the term type.

Keywords: Relation extraction · Sentiment analysis · Attention-based models

1 Introduction

Classifying relations between entities mentioned in texts remains one of the popular tasks in natural language processing (NLP). The sentiment attitude extraction task aims to seek for positive/negative relations between objects expressed as named entities in texts [10]. Let us consider the following sentence as an example (named entities are underlined):

> "Meanwhile Moscow has repeatedly emphasized that its activity in the Baltic Sea is a response precisely to actions of NATO and the escalation of the hostile approach to Russia near its eastern borders"

In the example above, named entities «Russia» and «NATO» have the negative attitude towards each other with additional indication of other named entities. The complexity of the sentence structure is one of the greatest difficulties one encounters when dealing with the relation extraction task. Texts usually

The reported study was partially supported by RFBR, research project № 20-07-01059.

E. Métais et al. (Eds.): NLDB 2020, LNCS 12089, pp. 157–169, 2020.
https://doi.org/10.1007/978-3-030-51310-8_15

contain a lot of named entity mentions; a single opinion might comprise several sentences.

This paper is devoted to study of models for targeted sentiment analysis with attention. The intuition exploited in the models with attentive encoders is that not all terms in the context are relevant for attitude indication. The interactions of words, not just their isolated presence, may reveal the specificity of contexts with attitudes of different polarities. The primary contribution of this work is an application of attentive encoders based on (I) sentiment frames and attitude participants (features); (II) context itself. We conduct the experiments on the RuSentRel [7] collection. The results demonstrate that attentive models with CNN-based and over LSTM-based encoders result in 1.5–5.9% by $F1$ over models without attentive encoders.

2 Related Work

In previous works, various neural network approaches for targeted sentiment analysis were proposed. In [10] the authors utilize convolutional neural networks (CNN). Considering relation extraction as a three-scale classification task of contexts with attitudes in it, the authors subdivide each context into *outer* and *inner* (relative to attitude participants) to apply Piecewise-CNN (PCNN) [16]. The latter architecture utilizes a specific idea of *max-pooling* operation. Initially, this is an operation, which extracts the maximal values within each convolution. However, for relation classification, it reduces information extremely rapid and blurs significant aspects of context parts. In case of PCNN, separate max-pooling operations are applied to outer and inner contexts. In the experiments, the authors revealed a fast training process and a slight improvement in the PCNN results in comparison to CNN.

In [12], the authors proposed an attention-based CNN model for semantic relation classification [4]. The authors utilized the attention mechanism to select the most relevant context words with respect to participants of a semantic relation. The architecture of the attention model is a multilayer perceptron (MLP), which calculates the weight of a word in context with respect to the entity. The resulting AttCNN model outperformed several CNN and LSTM based approaches with 2.6–3.8% by F1-measure.

In [9], the authors experimented with attentive models in aspect-based sentiment analysis. The models were aimed to identify sentiment polarity of specific *targets* in context, which are characteristics or parts of an entity. Both targets and the context were treated as *sequences*. The authors proposed an interactive attention network (IAN), which establishes element relevance of one sequence with the other in two directions: targets to context, context to targets. The effectiveness of IAN was demonstrated on the SemEval-2014 dataset [13] and several biomedical datasets [1].

In [14,17], the authors experimented with self-based attention models, in which *targets* became adapted automatically during the training process. Comparing with IAN, the presence of targets might be unclear in terms of algorithms.

The authors considered the attention as context word quantification with respect to abstract targets. In [14], the authors brought a similar idea also onto the sentence level. The obtained hierarchical model was called as HAN.

3 Data and Lexicons

We consider sentiment analysis of Russian analytical articles collected in the RuSentRel corpus [8]. The corpus comprises texts in the international politics domain and contains a lot of opinions. The articles are labeled with annotations of two types: (I) the author's opinion on the subject matter of the article; (II) the attitudes between the participants of the described situations. The annotation of the latter type includes 2000 relations across 73 large analytical texts. Annotated sentiments can be only *positive* or *negative*. Additionally, each text is provided with annotation of mentioned named entities. Synonyms and variants of named entities are also given, which allows not to deal with the coreference of named entities.

In our study, we also use two Russian sentiment resources: the RuSentiLex lexicon [7], which contains words and expressions of the Russian language with sentiment labels and the RuSentiFrames lexicon [11], which provides several types of sentiment attitudes for situations associated with specific Russian predicates.

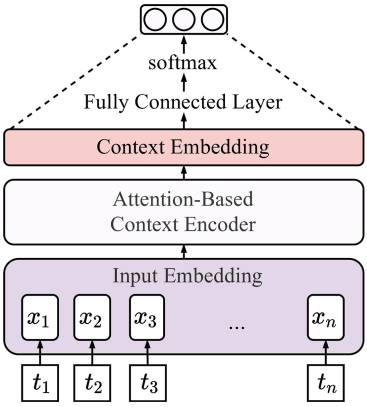
The RuSentiFrames[1] lexicon describes sentiments and connotations conveyed with a predicate in a verbal or nominal form [11], such as "осудить, улучшить, преувеличить" (to condemn, to improve, to exaggerate), etc. The structure of the frames in RuSentFrames comprises: (I) the set of predicate-specific roles; (II) frames dimensions such as the attitude of the author towards participants of the situation, attitudes between the participants, effects for participants. Currently, RuSentiFrames contains frames for more than 6 thousand words and expressions.

In RuSentiFrames, individual semantic roles are numbered, beginning with zero. For a particular predicate entry, `Arg0` is generally the argument exhibiting features of a Prototypical Agent, while `Arg1` is a Prototypical Patient or Theme [2]. In the main part of the frame, the most applicable for the current study is the polarity of `Arg0` with a respect to `Arg1` (`A0`→`A1`). For example, in case of Russian verb "одобрить" (to approve) the sentiment polarity `A0`→`A1` is positive.

4 Model

In this paper, the task of sentiment attitude extraction is treated as follows: given a pair of named entities, we predict a sentiment label of a pair, which could be positive, negative, or *neutral*. As the RuSentRel corpus provides opinions with positive or negative sentiment labels only (Sect. 3), we automatically added neutral sentiments for all pairs not mentioned in the annotation and co-occurred in the same sentences of the collection texts. We consider a *context* as a text fragment that is limited by a single sentence and includes a pair of named entities.

[1] https://github.com/nicolay-r/RuSentiFrames/tree/v1.0.



CONTEXT	Говорить о разделении кавказского региона из-за конфронтации **России**$_{obj}$ и **Турции**$_{subj}$ пока не приходится, хотя опасность есть.
	Talking about the separation of the Caucasus region due to the confrontation between **Russia**$_{obj}$ and **Turkey**$_{subj}$ is not necessary, although there is a danger.
TERMS	Talking about the separation of the **E** due to the **confrontation** between **E**$_{obj}$ and **E**$_{subj}$ is **not necessary** <COMMA> although there is a danger <DOT>

Fig. 1. (*left*) General, context-based 3-scale (positive, negative, neutral) classification model, with details on «Attention-Based Context Encoder» block in Sect. 4.1 and 4.2; (*right*) An example of a context processing into a sequence of terms; attitude participants («Russia», «Turkey») and other mentioned entities become masked; frames are bolded and optionally colored corresponding to the sentiment value of A0→A1 polarity.

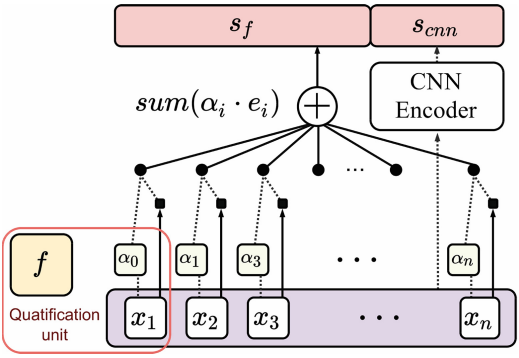
The general architecture is presented in Fig. 1 (left), where the sentiment could be extracted from the context. To present a context, we treat the original text as a sequence of terms $[t_1, \ldots, t_n]$ limited by n. Each term belongs to one of the following classes: ENTITIES, FRAMES, TOKENS, and WORDS (if none of the prior has not been matched). We use masked representation for attitude participants ($\underline{E}_{obj}$, $\underline{E}_{subj}$) and mentioned named entities (E) to prevent models from capturing related information.

To represent FRAMES, we combine a frame entry with the corresponding A0→A1 sentiment polarity value (and *neutral* if the latter is absent). We also invert sentiment polarity when an entry has "не" (not) preposition. For example, in Fig. 1 (right) all entries are encoded with the negative polarity A0→A1: "конфронтация" (confrontation) has a negative polarity, and "не приходится" (not necessary) has a positive polarity of entry "necessary" which is inverted due to the "not" preposition.

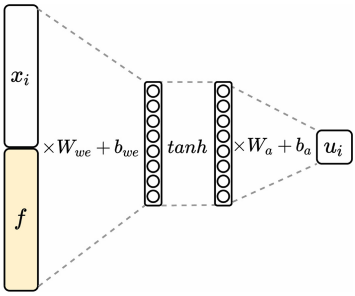
The TOKENS group includes: punctuation marks, numbers, url-links. Each term of WORDS is considered in a lemmatized[2] form. Figure 1 (right) provides a context example with the corresponding representation («TERMS» block).

To represent the context in a model, each term is embedded with a vector of fixed dimension. The sequence of embedded vectors $X = [x_1, \ldots, x_n]$ is denoted as *input embedding* ($x_i \in \mathbb{R}^m, i \in \overline{1..n}$). Sections 4.1 and 4.2 provide an encoder implementation in details. In particular, each encoder relies on input embedding and generates output *embedded context* vector s.

[2] https://tech.yandex.ru/mystem/.



(a) Context encoder architecture



(b) Quantification of term t_j as its embedded representation x_j relevance with respect to $f \in F$

Fig. 2. AttCNN neural network [6]

In order to determine a sentiment class by the embedded context s, we apply: (I) the hyperbolic tangent activation function towards s and (II) transformation through the *fully connected layer*:

$$r = W_r \cdot \tanh(s) + b_r \qquad W_r \in \mathbb{R}^{z \times c}, \quad b_r \in \mathbb{R}^c, \quad c = 3 \tag{1}$$

In Formula 1, W_r, b_r corresponds to hidden states; z correspond to the size of vector s, and c is a number of classes. Finally, to obtain an output vector of probabilities $o = \{\rho_i\}_{i=1}^c$, we use $softmax$ operation:

$$\rho_i = softmax(r_i) = \frac{\exp(r_i)}{\sum_{j=1}^{c} \exp(r_j)} \tag{2}$$

4.1 Feature Attentive Context Encoders

In this section, we consider *features* as a significant for attitude identification context terms, towards which we would like to quantify the relevance of each term in the context. For a particular context, we select embedded values of the (I) attitude participants ($\underline{E}_{obj}$, $\underline{E}_{subj}$) and (II) terms of the FRAMES group and create a set of features $F = [f_1, \ldots, f_k]$ limited by k.

MLP-Attention. Figure 2 illustrates a feature-attentive encoder with the quantification approach called Multi-Layer Perceptron [6]. In formulas 3–5, we describe the quantification process of a context embedding X with respect to a particular feature $f \in F$. Given an i'th embedded term x_i, we concatenate its representation with f:

$$h_i = [x_i, f] \qquad h_i \in \mathbb{R}^{2 \cdot m} \tag{3}$$

The quantification of the relevance of x_i with respect to f is denoted as $u_i \in \mathbb{R}$ and calculated as follows (see Fig. 2a):

$$u_i = W_a \left[\tanh(W_{we} \cdot h_i + b_{we})\right] + b_a \qquad W_{we} \in \mathbb{R}^{2 \cdot m \times \mathbf{h}_{\text{MLP}}}, \qquad W_a \in \mathbb{R}^{\mathbf{h}_{\text{MLP}}} \quad (4)$$

In Formula 4, W_{we} and W_a correspond to the weight and attention matrices respectively, and $\mathbf{h}_{\text{MLP}}$ corresponds to the size of the hidden representation in the weight matrix. To deal with normalized weights within a context, we transform quantified values u_i into probabilities α_i using $softmax$ operation (Formula 2). We utilize Formula 5 to obtain attention-based context embedding $\hat{s}$ of a context with respect to feature f:

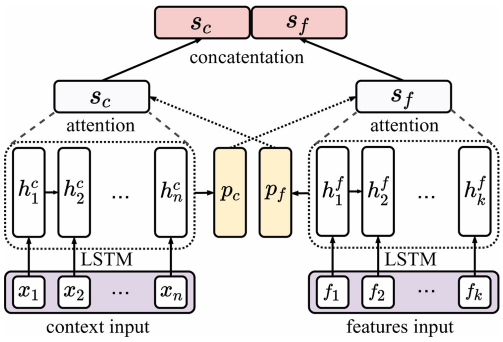
$$\hat{s} = \sum_{i=1}^{n} x_i \cdot \alpha_i \qquad \hat{s} \in \mathbb{R}^m \quad (5)$$

Applying Formula 5 towards each feature $f_j \in F$, $j \in \overline{1..k}$ results in vector $\{\hat{s}_j\}_{j=1}^{k}$. We use *average-pooling* to transform the latter sequence into single averaged vector $s_f = \hat{s}_j / [\sum_{j=1}^{k} \hat{s}_j]$.

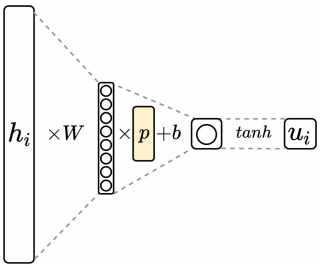
We also utilize a CNN-based encoder (Fig. 2b) to compete the context representation $s_{cnn} \in \mathbb{R}^{\mathbf{c}}$, where $\mathbf{c}$ is related to convolutional filters count [10]. The resulting context embedding vector s (size of $z = m + \mathbf{c}$) is a concatenation of s_f and s_{cnn}.

IAN. As a context encoder, a Recurrent Neural Network (RNN) model allows treating the context $[t_1, \ldots, t_n]$ as a sequence of terms to generate a hidden representation, enriched with features of previously appeared terms. In comparison with CNN, the application of RNN allows keeping a history of the whole sequence while CNN-based encoders remain limited by the window size. The application of RNN towards a context and certain features appeared in it – is another way how the correlation of these both factors could be quantitatively measured [9].

Figure 3a illustrates the IAN architecture attention encoder. The input assumes separated sequences of embedded terms X and embedded features F. To learn the hidden term semantics for each input, we utilize the LSTM [5] recurrent neural network architecture, which addresses learning long-term dependencies by avoiding gradient vanishing and expansion problems. The calculation h_t of t'th embedded term x_t based on prior state h_{t-1}, where the latter acts as a parameter of auxiliary functions [5]. The application of LSTM towards the input sequences results in $[h_1^c, \ldots, h_n^c]$ and $[h_1^f, \ldots, h_k^f]$, where $h_i^c, h_j^f \in \mathbb{R}^{\mathbf{h}}$ ($i \in \overline{1..n}$, $j \in \overline{1..k}$) and $\mathbf{h}$ is the size of the hidden representation. The quantification of input sequences is carried out in the following directions: (I) feature representation with respect to context, and (II) context representation with respect to features. To obtain the representation of a hidden sequence, we utilize *average-pooling*. In Fig. 3a, p_f and p_c denote a hidden representation of features and context respectively. Figure 3b illustrates the quantification computation of a hidden state h_t with respect to p:



(a) Context encoder architecture



(b) Quantification of term t_i based on related hidden state h_i with respect to pooled representation $p \in \mathbb{R}^{\mathbf{h}}$

Fig. 3. Interactive Attention Network (IAN) [9]

$$\begin{aligned} u_i^c &= \tanh(h_i^c \cdot W_f \cdot p_f + b_f) \qquad & W_f \in \mathbb{R}^{\mathbf{h}\times\mathbf{h}}, \;\; b_f \in \mathbb{R}, \;\; i \in \overline{1..n} \\ u_j^f &= \tanh(h_j^f \cdot W_c \cdot p_c + b_c) \qquad & W_c \in \mathbb{R}^{\mathbf{h}\times\mathbf{h}}, \;\; b_c \in \mathbb{R}, \;\; j \in \overline{1..k} \end{aligned} \tag{6}$$

In order to deal with normalized weight vectors α_i^f and α_j^c, we utilize the $softmax$ operation for u^f and u^c respectively (Formula 2). The resulting context vector s (size of $z = 2 \cdot \mathbf{h}$) is a concatenation of weighted context s_c and features s_f representations:

$$s_c = \sum_{i=1}^{n} \alpha_i^c \cdot h_i^c \quad s_f = \sum_{j=1}^{k} \alpha_j^f \cdot h_j^f \tag{7}$$

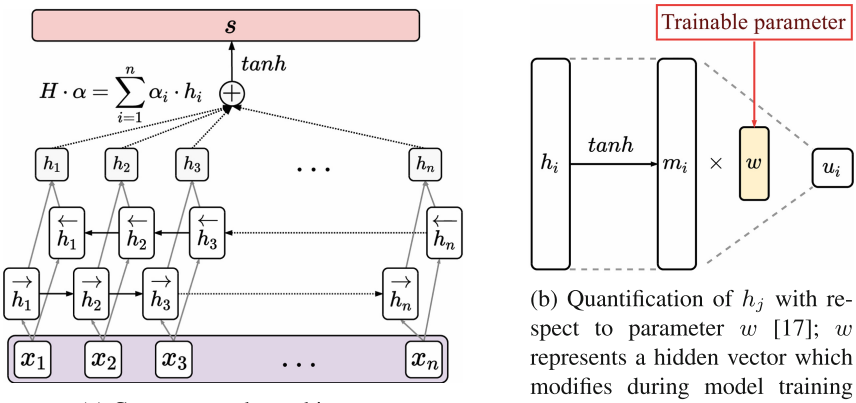
4.2 Self Attentive Context Encoders

In Sect. 4.1 the application of attention in context embedding fully relies on the sequence of predefined features. The quantification of context terms is performed towards each feature. In turn, the *self-attentive* approach assumes to quantify a context with respect to an abstract parameter. Unlike quantification methods in feature-attentive embedding models, here the latter is replaced with a hidden state (parameter w, see Fig. 4b), which modified during the training process.

Figure 4a illustrates the bi-directional RNN-based self-attentive context encoder architecture. We utilize bi-directional LSTM (BiLSTM) to obtain a pair of sequences $\overrightarrow{h}$ and $\overleftarrow{h}$ $(\overrightarrow{h_i}, \overleftarrow{h_i} \in \mathbb{R}^{\mathbf{h}})$. The resulting context representation $H = [h_1, \ \ldots, \ h_n]$ is composed as the concatenation of bi-directional sequences elementwise: $h_i = \overrightarrow{h_i} + \overleftarrow{h_i}$, $i \in \overline{1..n}$. The quantification of hidden term representation $h_i \in \mathbb{R}^{2\cdot\mathbf{h}}$ with respect to $w \in \mathbb{R}^{2\cdot\mathbf{h}}$ is described in formulas 8–9 and illustrated in Fig. 4b.

$$m_i = \tanh(h_i) \tag{8}$$

$$u_i = m_i^T \cdot w \tag{9}$$



(a) Context encoder architecture

(b) Quantification of h_j with respect to parameter w [17]; w represents a hidden vector which modifies during model training process

Fig. 4. Attention-based bi-directional LSTM neural network (Att-BLSTM) [17]

We apply the $softmax$ operation towards u_i to obtain vector of normalized weights $\alpha \in \mathbb{R}^n$. The resulting context embedding vector s (size of $z = 2 \cdot \mathbf{h}$) is an activated weighted sum of each parameter of context hidden states:

$$s = tanh(H \cdot \alpha) \tag{10}$$

5 Model Details

Input Embedding Details. We provide embedding details of context term groups described in Sect. 4. For WORDS and FRAMES, we look up for vectors in precomputed and publicly available model[3] M_{word} based on news articles with window size of 20, and vector size of 1000. Each term that is not presented in the model we treat as a sequence of *parts* (n-grams) and look up for related vectors in M_{word} to complete an averaged vector. For a particular part, we start with a trigram ($n = 3$) and decrease n until the related n-gram is found. For masked entities (E, $\underline{E}_{obj}$, $\underline{E}_{subj}$) and TOKENS, each element embedded with a randomly initialized vector with size of 1000.

Each context term has been additionally expanded with the following parameters:

- Distance embedding [10] ($v_{\text{D-}obj}$, $v_{\text{D-}subj}$) – is vectorized distance in terms from attitude participants of entry pair ($\underline{E}_{obj}$ and $\underline{E}_{subj}$ respectively) to a given term;
- Closest to synonym distance embedding ($v_{\text{SD-}obj}$, $v_{\text{SD-}subj}$) is a vectorized absolute distance in terms from a given term towards the nearest entity, synonymous to $\underline{E}_{obj}$ and $\underline{E}_{subj}$ respectively;

[3] http://rusvectores.org/static/models/rusvectores2/news_mystem_skipgram_1000_20_2015.bin.gz.

– Part-of-speech embedding (v_{POS}) is a vectorized tag for WORDS (for terms of other groups considering «unknown» tag);
– A0→A1 polarity embedding ($v_{A0 \rightarrow A1}$) is a vectorized «positive» or «negative» value for frame entries whose description in RuSentiFrames provides the corresponding polarity (otherwise considering «neutral» value); polarity is inverted when an entry has "НЕ" (not) preposition.

Training. This process assumes hidden parameter optimization of a given model. We utilize an algorithm described in [10]. The input is organized in minibatches, where minibatch yields of l *bags*. Each bag has a set of t pairs $\langle X_j, y_j \rangle_{j=1}^{t}$, where each pair is described by an input embedding X_j with the related label $y_j \in \mathbb{R}^c$. The training process is iterative, and each iteration includes the following steps:

1. Composing a minibatch of l bags of size t;
2. Performing forward propagation through the network which results in a vector (size of $q = l \cdot t$) of outputs $o_k \in \mathbb{R}^c$;
3. Computing cross entropy loss for output: $L_k = \sum_{j=1}^{c} \log p(y_i | o_{k,j}; \theta),\ k \in \overline{1..q}$;
4. Composing cost vector $\{cost_i\}_{i=1}^{l}$, $cost_i = \max\left[L_{(i-1)\cdot t} \,..\, L_{i \cdot t}\right)$ to update hidden variables set; $cost_i$ is a maximal loss within i'th bag;

Parameters Settings. The minibatch size (l) is set to 2, where contexts count per bag t is set to 3. All the sentences were limited by 50 terms. For embedding parameters ($v_{\text{D-}obj}$, $v_{\text{D-}subj}$, $v_{\text{SD-}obj}$, $v_{\text{SD-}subj}$, v_{POS}, $v_{A0 \rightarrow A1}$), we use randomly initialized vectors with size of 5. For CNN and PCNN context encoders, the size of convolutional window and filters count ($\mathbf{c}$) were set to 3 and 300 respectively. As for parameters related to sizes of hidden states in Sect. 4: $\mathbf{h}_{\text{MLP}} = 10$, $\mathbf{h} = 128$. For feature attentive encoders, we keep frames in order of their appearance in context and limit k by 5. We utilize the AdaDelta optimizer with parameters $\rho = 0.95$ and $\epsilon = 10^{-6}$ [15]. To prevent models from overfitting, we apply *dropout* towards the output with keep probability set to 0.8. We use Xavier weight initialization to setup initial values for hidden states [3].

6 Experiments

We conduct experiments with the RuSentRel[4] corpus in following formats:

1. Using 3-fold cross-validation (CV), where all folds are equal in terms of the number of sentences;
2. Using predefined TRAIN/TEST separation[5].

[4] https://github.com/nicolay-r/RuSentRel/tree/v1.1.
[5] https://miem.hse.ru/clschool/results.

Table 1. Three class context classification results by $F1$ measure (RuSentRel dataset); Columns from left to right: (I) average value in CV-3 experiment ($F1_{avg}$) with results on each split ($F1^{i}_{cv},\ i \in \overline{1..3}$); (II) results on TRAIN/TEST separation ($F1_{\text{TEST}}$)

Model	$F1_{avg}$	$F1^{1}_{cv}$	$F1^{2}_{cv}$	$F1^{3}_{cv}$	$F1_{\text{TEST}}$
Att-BLSTM	**0.314**	**0.35**	0.27	**0.32**	**0.35**
Att-BLSTM$^{z\text{-}yang}$	0.292	0.33	0.25	0.30	0.33
BiLSTM	0.286	0.32	0.26	0.28	0.34
IAN$_{ef}$	0.289	0.31	0.28	0.27	0.32
IAN$_{ends}$	0.286	0.31	0.26	0.29	0.32
LSTM	0.284	0.28	0.27	0.29	0.32
PCNN$_{att\text{-}ends}$	0.297	0.32	**0.29**	0.28	**0.35**
PCNN$_{att\text{-}ef}$	0.289	0.31	0.25	0.31	0.31
PCNN	0.285	0.29	0.27	0.30	0.32

In order to evaluate and assess attention-based models, we provide a list of baseline models. These are independent encoders described in Sects. 4.1 and 4.2: PCNN [10], LSTM, BiLSTM. In case of models with feature-based attentive encoders (IAN$_*$, PCNN$_*$) we experiment with following feature sets: attitude participants only (*att-ends*), and frames with attitude participants (*att-ef*). For self-based attentive encoders we experiment with Att-BLSTM (Sect. 4.2) and Att-BLSTM$^{z\text{-}yang}$ – is a bi-directional LSTM model with word-based attentive encoder of HAN model [14].

Table 1 provides related results. For evaluating models in this task, we adopt macroaveraged F1-score ($F1$) over documents. F1-score is considered averaging of the positive and negative class. We measure $F1$ on train part every 10 epochs. The number of epochs was limited by 150. The training process terminates when $F1$ on train part becomes greater than 0.85. Analyzing $F1_{\text{TEST}}$ results it is quite difficult to demarcate attention-based models from baselines except Att-BLSTM and PCNN$_{att\text{-}ends}$. In turn, average results by $F1$ in the case of CV-3 experiments illustrate the effectiveness of attention application. The average increase in the performance of such models over related baselines is as follows: 1.4% (PCNN$_*$), 1.2% (IAN$_*$), and 5.9% (Att-BLSTM, Att-BLSTM$^{z\text{-}yang}$) by $F1$. The greatest increase in 9.8% by $F1$ is achieved by Att-BLSTM model.

7 Analysis of Attention Weights

According to Sects. 4.1 and 4.2, attentive embedding models perform the quantification of terms in the context. The latter results in the probability distribution of weights[6] across the terms mentioned in a context.

[6] We consider and analyze only context weights in case of IAN models.

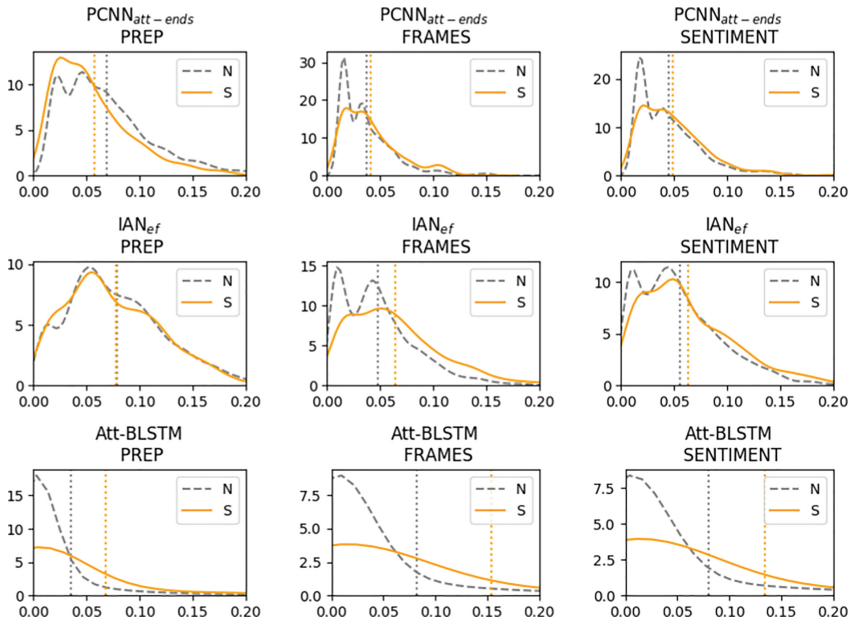


Fig. 5. Kernel density estimations (KDE) of context-level weight distributions of term groups (from left to right: PREP, FRAMES, SENTIMENT) across *neutral* (N) and *sentiment* (S) context sets for models: PCNN$_{att\text{-}ef}$, IAN$_{ef}$, ATT-BLSTM; the probability range (x-axis) scaled to $[0, 0.2]$; vertical lines indicate expected values of distributions

We utilize the TEST part of the RuSentRel dataset (Sect. 6) for analysis of weight distribution of FRAMES group, declared in Sect. 4, across all input contexts. We also introduce two extra groups utilized in the analysis by separating the subset of WORDS into prepositions (PREP) and terms appeared in RuSentiLex lexicon (SENTIMENT) described in Sect. 3.

The *context-level weight* of a group is a weighted sum of terms which both appear in the context and belong the corresponding term group. Figure 5 illustrates the weight distribution plots, where the models are organized in rows, and the columns correspond to the term groups. Each plot combines distributions of context-levels weights across:

- **Neutral contexts** – contexts, labeled as **neutral**;
- **Sentiment contexts** – contexts, labeled with **positive or negative** labels.

In Fig. 5 and further, the distribution of context-level weights across neutral («N» in legends) and sentiment contexts («S» in legends) denoted as ρ^g_N and ρ^g_S respectively. The rows in Fig. 5 correspond to the following models: (1) PCNN$_{att\text{-}ef}$, (2) IAN$_{ef}$, (3) ATT-BLSTM. Analyzing prepositions (column 1) it is possible to see the lack of differences in quantification between the ρ^{PREP}_N and ρ^{PREP}_S contexts in the case of the models (1) and (2). Another situation is in case of the model (3), where related terms in sentiment contexts

are higher quantified than in neutral ones. FRAMES and SENTIMENT groups are slightly higher quantified in sentiment contexts than in neutral one in the case of models (1) and (2), while (3) illustrates a significant discrepancy.

Overall, model ATT-BLSTM stands out among others both in terms of results (Sect. 6) and it illustrates the greatest discrepancy between ρ_N and ρ_S across all the groups presented in the analysis (Fig. 5). We assume that the latter is achieved due to the following factors: (I) application of bi-directional LSTM encoder; (II) utilization of a single trainable vector (w) in the quantification process (Fig. 4b) while the models of other approaches (ATTCNN, IAN, and ATT-BLSTM$^{z\text{-}yang}$) depend on fully-connected layers. Figure 6 shows examples of those sentiment contexts in which the weight distribution is the largest among the FRAMES group. These examples are the case when both frame and attention masks convey context meaning.

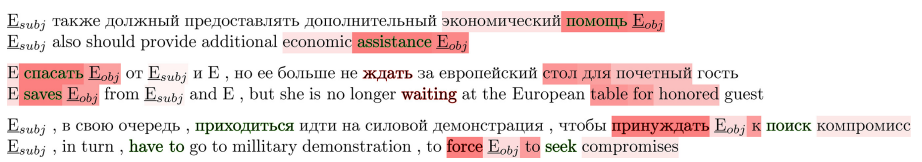


Fig. 6. Weight distribution visualization for model ATT-BLSTM on sentiment contexts; for visualization purposes, weight of each term is normalized by maximum in context

8 Conclusion

In this paper, we study the attention-based models, aimed to extract sentiment attitudes from analytical articles. The described models should classify a context with an attitude mentioned in it onto the following classes: positive, negative, neutral. We investigated two types of attention embedding approaches: (I) feature-based, (II) self-based. We conducted experiments on Russian analytical texts of the RuSentRel corpus and provide the analysis of the results. According to the latter, the advantage of attention-based encoders over non-attentive was shown by the variety in weight distribution of certain term groups between sentiment and non-sentiment contexts. The application of attentive context encoders illustrates the classification improvement in 1.5–5.9% range by $F1$.

References

1. Alimova, I., Solovyev, V.: Interactive attention network for adverse drug reaction classification. In: Ustalov, D., Filchenkov, A., Pivovarova, L., Žižka, J. (eds.) AINL 2018. CCIS, vol. 930, pp. 185–196. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01204-5_18
2. Dowty, D.: Thematic proto-roles and argument selection. Language **67**(3), 547–619 (1991)

3. Glorot, X., Bengio, Y.: Understanding the difficulty of training deep feedforward neural networks. In: Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, pp. 249–256 (2010)
4. Hendrickx, I., et al.: Semeval-2010 task 8: multi-way classification of semantic relations between pairs of nominals. In: Proceedings of the Workshop on Semantic Evaluations: Recent Achievements and Future Directions, pp. 94–99 (2009)
5. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997)
6. Huang, X., et al.: Attention-based convolutional neural network for semantic relation extraction. In: Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers, pp. 2526–2536 (2016)
7. Loukachevitch, N., Levchik, A.: Creating a general Russian sentiment lexicon. In: Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016), pp. 1171–1176 (2016)
8. Loukachevitch, N., Rusnachenko, N.: Extracting sentiment attitudes from analytical texts. In: Proceedings of International Conference on Computational Linguistics and Intellectual Technologies Dialogue-2018 (arXiv:1808.08932), pp. 459–468 (2018)
9. Ma, D., Li, S., Zhang, X., Wang, H.: Interactive attention networks for aspect-level sentiment classification. arXiv preprint arXiv:1709.00893 (2017)
10. Rusnachenko, N., Loukachevitch, N.: Neural network approach for extracting aggregated opinions from analytical articles. In: Manolopoulos, Y., Stupnikov, S. (eds.) DAMDID/RCDL 2018. CCIS, vol. 1003, pp. 167–179. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-23584-0_10
11. Rusnachenko, N., Loukachevitch, N., Tutubalina, E.: Distant supervision for sentiment attitude extraction. In: Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2019) (2019)
12. Shen, Y., Huang, X.: Attention-based convolutional neural network for semantic relation extraction. In: Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers, pp. 2526–2536 (2016)
13. Wagner, J., et al.: DCU: aspect-based polarity classification for SemEval task 4 (2014)
14. Yang, Z., Yang, D., Dyer, C., He, X., Smola, A., Hovy, E.: Hierarchical attention networks for document classification. In: Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics, pp. 1480–1489 (2016)
15. Zeiler, M.D.: ADADELTA: an adaptive learning rate method. arXiv preprint arXiv:1212.5701 (2012)
16. Zeng, D., Liu, K., Chen, Y., Zhao, J.: Distant supervision for relation extraction via piecewise convolutional neural networks. In: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, pp. 1753–1762 (2015)
17. Zhou, P., et al.: Attention-based bidirectional long short-term memory networks for relation classification. In: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, vol. 2, pp. 207–212 (2016)