Improving Neural Network Models for Natural Language Processing in Russian with Synonyms

Ruslan Galinsky, Anton Alekseev
Steklov Mathematical Institute at St. Petersburg
St. Petersburg, Russia
{galinskyifmo, anton.m.alexeyev}@gmail.com

Sergey I. Nikolenko
Steklov Mathematical Institute at St. Petersburg, Russia
Kazan Federal University, Kazan, Russia
sergey@logic.pdmi.ras.ru

***Abstract*—Recent advances in deep learning for natural language processing achieve and improve over state of the art results in many natural language processing tasks. One problem with neural network models, however, is that they require large datasets, including large labeled datasets for the corresponding problems. In this work, we suggest a data augmentation method based on extending a given dataset with synonyms for the words appearing there. We apply this approach to the morphologically rich Russian language and show improvements for modern neural network NLP models on standard tasks such as sentiment analysis.**

I. Introduction

Deep learning for natural language processing is a burgeoning field that brings new advances every month, perhaps even every week. Although it started with more or less standard architectures (recurrent and convolutional neural networks), it is beginning to branch out into several quite different directions (from recursive networks for syntactic parsing to attention-based models for machine translation and memory networks for question answering). Moreover, by now deep learning has become very much an engineering field: thanks to the automatic differentiation libraries such as *Theano* [7] or *TensorFlow* [1] and libraries such as *Keras* [11] or *Lasagne* [14] that implement various neural network components, layers, and optimization algorithms, experimenting with new neural architectures in practice has transformed from a tedious error-prone affair into a relatively easy and exciting process.

However, large-scale neural network models require very large datasets to be trained efficiently. While it is usually easy to collect large unlabeled text datasets, it may be hard to collect large datasets for a specific problem such sentiment analysis, syntactic parsing, machine translation, and so on.

In computer vision, it is common practice to augment the input datasets by slight changes in the input images. Computer vision yields itself very easily to such modifications: if we slightly crop, shift, or contract an image, change lighting conditions or downsample to reduce resolution, the objects on the image will remain the same, and the recognition target can be reused. This is not even denoising as in denoising autoencoders, it is simply new training data for free. These augmentation procedures are used in most modern computer vision models; see, e.g., [24], [34] and references therein.

In a way, computer vision is lucky to have an almost unlimited source of new training samples but in natural language processing one cannot simply change a word at random and assume that the “big picture” will remain exactly the same. Ideally, we might use human paraphrases but they are impossible to obtain in the necessary quantities. Zhang et al. [36] propose a straightforward idea for such data augmentation: use a human-generated standard thesaurus (from WordNet in their case) and replace some words at random with their direct synonyms. They report improved results with this augmentation, but it appears that there might be other transformations helpful for NLP data augmentation, and this problem may warrant further study.

In this work, we modify and apply this scheme to the Russian language; besides, we propose and evaluate another data augmentation scheme based on extending user reviews (in a sentiment analysis task) with additional adjectives. The paper is organized as follows. In Section II, we discuss an important idea for natural language processing based on deep learning, namely moving from word-level embeddings such as *word2vec* to character-level models. Section III discusses in detail the data augmentation procedures we evaluated. Section IV shows experimental results that validate that augmentation based on synonyms does improve sentiment analysis results, and Section V concludes the paper.

II. Character-level models

Recent advances in distributed word representations have made it into a method of choice for modern natural language processing [15]. Distributed word representations are models that map each word occurring in the dictionary to a Euclidean space, attempting to capture semantic relationships between the words as geometric relationships in the Euclidean space. In a classical word embedding model, one first constructs a vocabulary with one-hot representations of individual words, where each word corresponds to its own dimension, and then trains representations for individual words starting from there, basically as a dimensionality reduction problem. For this purpose, researchers have usually employed a model with one hidden layer that attempts to predict the next word based on a window of several preceding words. Then representations learned at the hidden layer are taken to be the word’s features. The modern field of word embeddings started with the work [5], subsequently extended in [6]. Extending previous work on statistical language models that were usually based on word n-grams [9], [10], [16], [21], Bengio et al. proposed the idea of *distributed word representations*, the idea of word embeddings was applied back to language modeling, e.g., in [28], [29], [31], and then, starting from the works of

ISBN 978-952-68397-8-3 (PDF)

Mikolov et al. [27], [30], word representations have been applied for numerous natural language processing problems, including text classification, extraction of sentiment lexicons, part-of-speech tagging, syntactic parsing and so on.

To train distributed word representations, one first constructs a vocabulary with one-hot representations of individual words (where each word is represented with a vector of size equal to vocabulary size with a single 1) and then trains representations for individual words starting from there, basically as a dimensionality reduction problem. For this purpose, researchers have usually employed a model with one hidden layer that attempts to predict the next word based on a window of several preceding words. There exist two most commonly used models for word embeddings, both introduced in [27] and based on previous work on neural probabilistic language models [5]: *Continuous Bag-of-Words* (CBOW), which tries to reconstruct words from their contexts, and *skip-gram*, which operates inversely, reconstructing word contexts from the words themselves. Then representations learned at the hidden layer are taken to be the word's features; this approach has been applied, for instance in the Polyglot system developed in 2013 [3] and in other methods of learning distributed word representations [33]. A recent study on the performance of various vector space models for word semantic similarity evaluation [32] demostrates that compositions of models such as GloVe and Word2Vec as well as unsupervised one-model approaches show reasonable results for the Russian language.

However, word embeddings as introduced in [27] and other works suffer from some conceptual flaws:

(1) first, the vectors trained for every word are completely independent; this means that we cannot really reuse our knowledge about one word to get an understanding for another, like people do; in particular, in morphology-rich languages like Russian, each word comes with a plethora of different morphological forms, various derivative words in other parts of speech, derivative words formed by prefixes and suffixes and so on; a human being understands all these derivative words immediately after he or she understands the basic word but a word embedding model would have to either cluster all of them together in the same vector or obtain a sufficient quantity of usage examples for every form, which probably will not happen;
(2) second, the same applies to out-of-vocabulary words: a word embedding cannot be extended to new words without a reasonably sized set of usage examples while a human being can extrapolate the meaning from the form of a word; e.g., you may never have encountered the word *polydistributional* (it had been getting only 48 results on Google before we started using it as an example) but you already have a pretty good idea of what it means;
(3) third, as a practical consideration word embedding models may grow large for large vocabularies; although applying a trained model is very fast (it is just lookup to the table of word vectors), either the model has to be stored in memory or access will still be slow.

These problems lead to the idea of *character-level representations*: what if we descend down to the most basic level of written speech and train word embeddings that take into account the actual characters that comprise a word. This set of approaches is highly relevant for the proposed project as Russian, being a very morphology-rich language, would probably benefit greatly from such approaches.

First attempts at this problem involved decomposing a word into *morphemes*, the smallest units of meaning in written language [8], [26], [35]. If morphemes were available explicitly they would indeed be a perfect building block for a low-level word representation model since they are precisely what carries the meaning. However, in practice morphemes are not immediately evident from a word, and one has to rely on morphological analyzers that work imperfectly and basically introduce the need to train a separate morphology model, so the problem only shifts to that model.

In [25], Ling et al. present a *character to word* (C2W) model for learning word embeddings based on bidirectional LSTMs [17], [18]. A bidirectional LSTM basically consists of two LSTMs, forward and backward, and the final representation is a linear combination of their states (again with weights to be trained as part of the model). Ling et al. report state of the art results in language modeling (in terms of perplexity) and part-of-speech tagging, especially for morphology-rich languages.

Note that applying a character model is relatively expensive, and it would slow down applications significantly if one had to run a bidirectional LSTM for every word. Fortunately, since the C2W model depends only on the characters it is easy to just store the representations of common words in memory, recalculating them only for rare words; this way, one can strike a proper balance between memory and computational time.

New developments have also begun to appear in character-level models. For instance, a very recent work by Chung et al. [12] explore the possibilities of constructing a machine translation model which is not based exclusively on word embeddings but augments it with a character-level model, producing a unified character-level model with machine translation, achieving state-of-the-art results. Although these results do not significantly outperform word-based approaches, the work [12] clearly shows that it is possible to construct character-level models for machine translation, and they do not break down as they might because translations in characters are much longer than translations in words. Finally, recent work on character-level models for morphologically rich languages has introduced morphological smoothing that could model the morphological variation in the word embedding space [13] and explicit representations of morphological features for reinflection [19].

Character-level models are especially important for developing NLP models for the Russian language for two main reasons. First, they are very well suited for languages with rich morphology, such as Russian; Russian contains plenty of words that are tightly linked with each other (have the same root), and shades of meaning are distinguished with morphemes. It would be obviously very wasteful to treat all of them as separate words. One can use available morphological analyzers to connect different forms of the same word (we do so in auxiliary steps of this work too), but then one has to either disregard morphological data, which loses meaningful information, or again treat different forms of a word as different words. Second, character-level models are also well suited for studies of user-generated texts such as user reviews, social network statuses, blog posts, and the like; user-generated texts abound with typos, intentional misspellings, word spelling variations, and so on, which are immediately recognized by human readers but are impossible to pick up for a word-based model. This work is one of the first steps towards a general-purpose character-level model for the Russian language.

In our experiments, we used a character-level model similar to the one presented in [36], where Zhang et al. develop a natural approach to constructing character-level representations based on convolutional neural networks. They report significant improvements for standard text classification problems. They also suggest a straightforward way for data augmentation: replacing a word with its direct synonym. However, for Russian and other morphologically rich languages this scheme is harder to apply as the new word has to match the syntax as well as the semantics of the old word. We are not aware of previous work on such data augmentation for Russian; other data augmentation approaches have included, e.g., anaphora resolution as a preprocessing technique to improve the word embeddings [23].

III. Data augmentation approaches

A. Replacing words with their synonyms

To achieve data augmentation with synonyms, we begin with collecting and filtering a set of pairs of synonymous words. We begin with publicly available dictionaries of synonyms (thesauri) for the Russian language, collected from online versions of dictionaries of synonyms [2], [4]. We also used a general frequency vocabulary of the Russian language, running a preliminary filter to exclude archaic or very rare words.

At the data augmentation stage, we use an explicit morphological analyzer *pymorphy* [22]; naturally, the use of an automated analyzer introduces a certain share of errors but the errors are rare enough to still lead to overall improvement. First, we use *pymorphy* to find the part of speech and other morphological data for all words and leave only nouns and adjectives. Then we take the synonyms to have the same gender: masculine noun with masculine noun and so on.

In thesauri, it often happens that some words are more general, and others are their special cases; in this case, it may be incorrect to replace the general word with a more specific, less abstract word. For example, it is almost always correct to replace *car* with *automobile* but not with *minivan*, although a thesaurus may mark *car* as a synonym for *minivan*. In real world thesauri, we will not be able to automatically find which one in an asymmetrical pair of synonyms is more general, so as the next filter we checked reflexivity: we only use w_1 as a synonym for w_2 if both w_1 is marked as a synonym for w_2 in the thesaurus and w_2 is marked as a synonym of w_1 in the thesaurus.

At this point, we have a set S of unordered pairs of synonyms that we assume to be safe to use for replacement.

Next, we go through the input text and feed it through *pymorphy*. The analyzer outputs morphological features for each word. For every word w, we:

- remember its morphological features and take its base form w_0 as suggested by *pymorphy*;
- look for the synonyms of the base form w_0 in the set of synonyms S, getting the set of synonyms $S_w = \{w' \mid (w_0, w') \in W\}$;
- sample a synonym w_0' from S_w according to a multinomial distribution with probabilities proportional to the word frequencies (overall frequencies in the Russian language).

Note that at the sampling stage, we can either include the word w_0 itself in S_w, regarding it as its own synonym, or leave it out. Our experiments show that it is beneficial to include the word w_0 itself in S_w, sometimes leaving the word in place even if it does have synonyms in S. This turns out to be important in cases when the word is very frequent, and synonyms are rare and unlikely to appear so it is better to leave it in place.

Then we use *pymorphy* to map the word w_0' back to the form used in the review and replace the original word w with the resulting form w'.

B. Reshuffling the words

Another straightforward technique for data augmentation is to reshuffle the words. The correct way to shuffle words would be to automatically construct parse trees from the sentences and then randomly change places of certain subtrees; the less rigid word order in Russian makes this approach attractive. However, in this work we only use a very simple and obviously incorrect approach of word reshuffling, basically turning it into a bag of words. Somewhat surprisingly, we will see in Section IV that even if we shuffle all words randomly, the resulting sentiment recognition quality does not change all that much.

C. Adding new adjectives

Experiments with reshuffling words in a review (we did not get significant reduction in quality from basically converting the review into a bag of words) suggest that we could try to generate “simulated reviews” by simply sampling suitable words. We tested this idea with an experiment on adding new adjectives and/or verbs since adjectives and verbs are usually the most characteristic words for sentiment evaluation (as our counting experiments shown below suggest).

For the new augmentation procedure, we have chosen to add new adjectives. For preprocessing, we collected the following statistics, again using *pymorphy* for part of speech tagging and lemmatization:

- count how many times a given (lemmatized) adjective occurs in the dataset both in positive and negative reviews (some of these results are discussed below and shown in Table II);
- count how many times a given adjective appears before or after a noun (we did not perform full syntactic parsing here, simply counted occurrences of noun-adjective and adjective-noun bigrams);
- count how many times a given adjective occurs next to a given noun.

TABLE I. DATASET STATISTICS

Dataset	Reviews		
	Positive	Negative	Total
Basic: torg.mail.ru + Restoclub	63088	35046	98134
Augmented with adjectives	126176	70092	196268
Augmented with synonyms	125523	69849	195372
Test dataset: TripAdvisor	26807	11075	37882

TABLE II. IMBALANCED WORDS IN VARIOUS PARTS OF SPEECH

Word		Counts				
Russian	English	Pos.	% pos.	Neg.	% neg.	Diff.
Adjectives						
замечательный	wonderful	5537	0.088	1153	0.033	0.055
огромный	huge	7251	0.115	2052	0.059	0.056
вежливый	polite	6853	0.109	1759	0.050	0.058
красивый	pretty	7921	0.126	2331	0.067	0.059
прекрасный	beautiful	6713	0.106	1620	0.046	0.060
...	...					
должный	must	3171	0.050	5249	0.150	-0.100
отвратительный	disgusting	332	0.005	2716	0.077	-0.072
ужасный	terrible	453	0.007	2746	0.078	-0.071
никакой	bad	4060	0.064	4616	0.132	-0.067
данный	this	3229	0.051	4111	0.117	-0.066
Nouns						
свадьба	wedding	4718	0.075	1244	0.035	0.039
атмосфера	atmosphere	6734	0.107	2317	0.066	0.041
площадь	area	4937	0.078	1153	0.033	0.045
храм	temple	4271	0.068	363	0.010	0.057
собор	cathedral	5045	0.080	599	0.017	0.063
...	...					
итог	total	3710	0.059	6349	0.181	-0.122
счёт	bill	3374	0.053	6063	0.173	-0.120
ответ	response	1846	0.029	5047	0.144	-0.115
том	volume	6017	0.095	7109	0.203	-0.107
фильм	movie	5461	0.087	6773	0.193	-0.107
Verbs						
помочь	help	3561	0.056	1245	0.036	0.021
отмечать	note	3133	0.050	975	0.028	0.022
посетить	visit	5801	0.092	2165	0.062	0.030
порадовать	gladden	5406	0.086	1546	0.044	0.042
доставить	deliver	5939	0.094	1804	0.051	0.043
...	...					
звонить	call	2111	0.033	5017	0.143	-0.110
вернуть	return	1345	0.021	4562	0.130	-0.109
стать	become	5450	0.086	6725	0.192	-0.106
позвонить	call	5148	0.082	6223	0.178	-0.096
говорить	speak	3078	0.049	4791	0.137	-0.088
Adverbs						
вовремя	timely	2966	0.047	550	0.016	0.031
удобно	conveniently	3787	0.060	977	0.028	0.032
отлично	excellently	4599	0.073	1065	0.030	0.043
приятно	pleasantly	7743	0.123	1834	0.052	0.070
обязательно	certainly	6939	0.110	1172	0.033	0.077
...	...					
вообще	generally	5386	0.085	7373	0.210	-0.125
потом	after	4395	0.070	5830	0.166	-0.097
почему	why	3022	0.048	5058	0.144	-0.096
более	more	5848	0.093	5982	0.171	-0.078
видимо	seemingly	1836	0.029	3725	0.106	-0.077

After these statistics have been connected, for the augmentation we go over the text of a given review, looking for nouns. If a noun w does not have an associated adjective (i.e., an adjective either before or after it), we perform the following procedure:

- sample whether to add an adjective to this noun based on statistics on how often w appears with and without adjectives;
- if an adjective should be added, sample which one to add from the multinomial distribution with probabilities proportional to the numbers of times different adjectives occur in positive and negative reviews next to this noun;
- then sample whether it should be added before or after the noun based on the corresponding statistic;
- then add the resulting adjective to the text.

After this augmentation procedure, we get reviews with additional adjectives that adhere to the dataset statistics and do indeed most often "make sense" for the corresponding words.

IV. EVALUATION

A. Datasets and basic statistics

For experimental evaluation, we have chosen the sentiment analysis problem since it is relatively easy to mine large train and test datasets for this classical NLP problem. To try to train for general sentiment rather than for a specific subject domain, we have collected our basic dataset from two very different sources: marketplace reviews from *torg.mail.ru* and restaurant reviews from *www.restoclub.ru*. The basic statistics are shown in Table I.

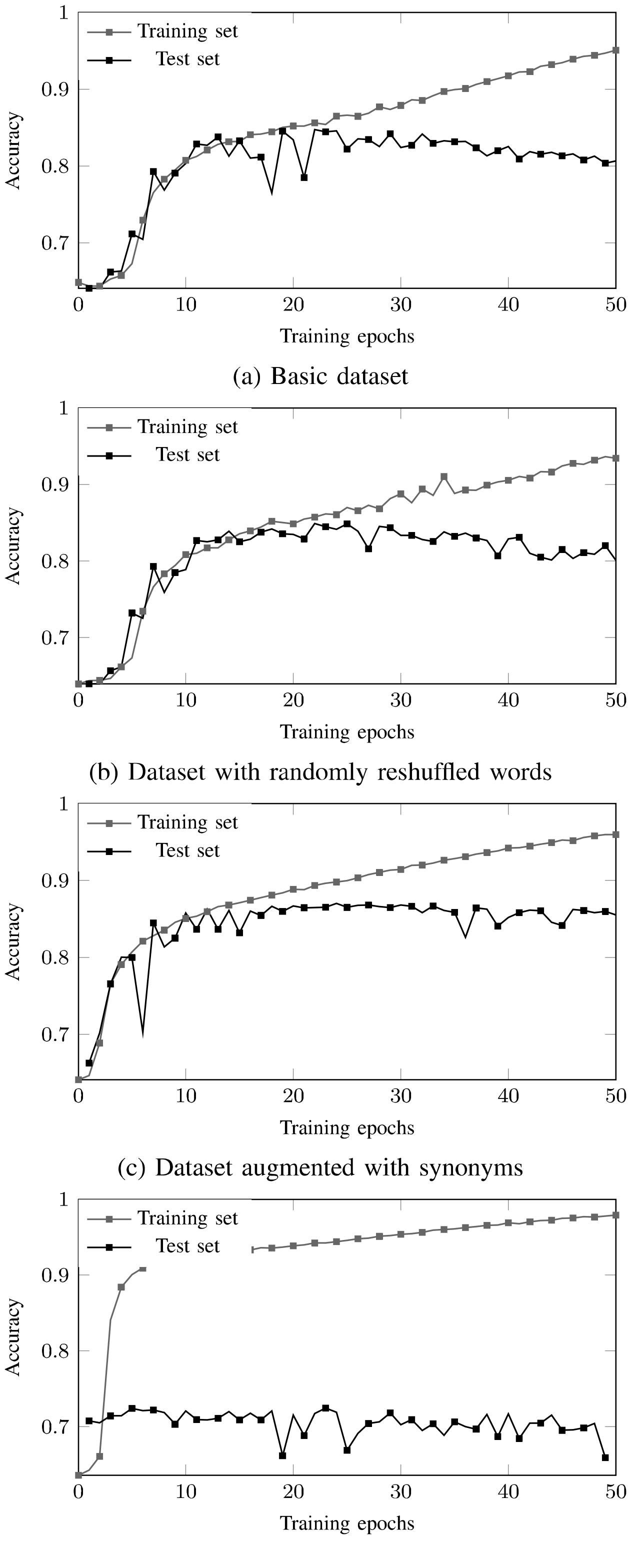
Next, we have applied the augmentation procedures described in detail in Section III to obtain two extended datasets: one augmented with additional adjectives as shown in Section III-C and another augmented with direct synonyms as shown in Section III-A. In each case, we have extended the basic dataset by approximately a factor of two, adding one modified review for each original one.

Besides, to test how well the resulting sentiment models transfer to a different domain, we have collected another, smaller dataset from a completely different source: hotel reviews from the *TripAdvisor* Web site. This dataset was never used in training, but we evaluated the quality of our models on it. Note, however, that results on the *TripAdvisor* dataset are expected to be significantly worse not only because the domain is different but also due to the properties of the *TripAdvisor* dataset itself: it has a different distribution of review scores, with about 90% of the reviews scoring five stars.

Another interesting piece of data is the number of occurrences of words in positive and negative reviews; in our experiments, it plays a role for data augmentation with adjectives and verbs as discussed in Section III-C. Table II shows the most imbalanced positive and negative words for various parts of speech; some entries represent lemmatization errors or confusion between different words but mostly they paint a reasonable picture. It is also clear that the most imbalanced (colored) words are adjectives and nouns.

B. Training the model

Our model was based on the *keras* [11] implementation of the model presented in [36] (https://github.com/johnb30/py_crepe). We have used the same topology: starting from character quantization with a simple 1-of-m encoding, the unprocessed text data is fed to a convolutional net with 6 convolutional layers, 3 fully connected layers, and 2 dropout modules between fully connected layers for regularization; we used 1024 units on the fully connected top layers. We have used the Adam optimizer [20] for training. All experiments were conducted on a single NVIDIA Titan X GPU. The training and test set errors for the basic dataset are shown on Fig. 1a.



(a) Basic dataset

(b) Dataset with randomly reshuffled words

(c) Dataset augmented with synonyms

(d) Dataset extended with adjectives

Fig. 1. Accuracy on training and test sets during training of the models in the study

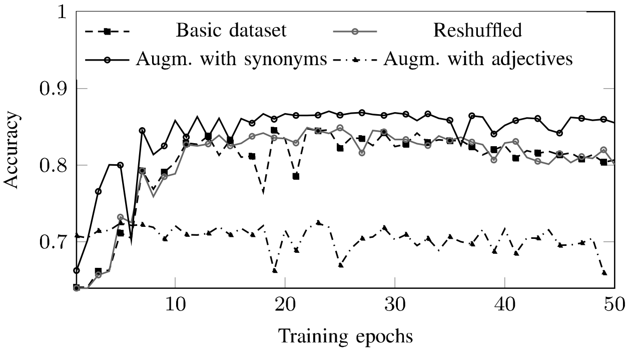


Fig. 2. A comparison of test set accuracies of all models in the study

TABLE III. EXPERIMENTAL RESULTS

Dataset	Best accuracy	
	Test set	TripAdvisor set
Basic dataset	0.8457	0.7163
Basic with reshuffled words	0.8445	0.7160
Augmented with adjectives	0.7241	0.5430
Augmented with synonyms	0.8700	0.7020

C. Word reshuffling

In this experiment, we have trained and applied the model to the basic dataset with all words in each review randomly reshuffled. Somewhat surprisingly, test set accuracy of the resulting model is virtually indistinguishable from the original, and the score on a separate TripAdvisor dataset from a completely different domain (see Table III) is also approximately the same as the original model. This indicates that, first, it might make sense to add new words to reviews even if they slightly violate grammatical rules because the grammar does not seem to matter much; and second, that the models still have a long way to go before they can achieve real understanding of sentiment since it does obviously depend on word order.

D. Augmented datasets

We have also trained and tested the model on augmented datasets, with synonyms and with additional adjectives. Fig. 1c shows training and test errors for the dataset augmented with synonyms, Fig. 1d, for the dataset augmented with adjectives, Fig. 2 compares the test set errors across all four experiments, and Table III summarizes the results.

The results on augmentation with synonyms were positive: we have seen significant improvements in both training and test set accuracy in our experiments. However, data augmentation with additional adjectives did not work, producing worse results than even the original dataset. This can be explained by overfitting: adding sentiment-heavy adjectives has resulted in a training set full with specific words that mark sentiment, so the model had trained to recognize these words and could not process the test set without this abundance qiute as well.

E. TripAdvisor experiment

We have also performed an additional experiment, evaluating the quality of the resulting models on a problem domain where they had not been trained, namely on hotel reviews from TripAdvisor. The accuracy of different models on this additional dataset is also shown in Table III. The results indicate that so far, the resulting sentiment models do not transfer easily from one domain to another: across all datasets, results on the test set are significantly worse, and the improvements from synonym-based data augmentation have disappeared. This indicates that general-purpose sentiment models are still subject for further work.

V. Conclusion

In this work, we have introduced and evaluated several different approaches to data augmentation for natural language processing in the context of character-level models. Our results show promise: it appears that even simple data augmentation with synonyms taken from common thesauri can yield significant improvements for common NLP problems such as sentiment analysis. We propose to use augmentation with synonyms as a tool to extend insufficiently large datasets; note that this tool is based on additional information from the thesauri of synonyms.

On the other hand, we have seen that not every augmentation is beneficial: an extension with extra adjectives turned out to produce worse results, probably due to overfitting. In further work, we plan to improve upon these augmentation approaches and produce state of the art character-level models for the Russian language.

Acknowledgments

This work was partially supported by the Russian Federation grant 14.Z50.31.0030 and by the 2016 grant "User Profiling Based on Distributed Word Representations" sponsored by Samsung Russia.

References

[1] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[2] N. Abramov. *Dictionary of Russian Synonyms and Synonymous Phrases*. Moscow: Russkie Slovari, 1999.

[3] R. Al-Rfou, B. Perozzi, and S. Skiena. Polyglot: Distributed word representations for multilingual nlp. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, pages 183–192, Sofia, Bulgaria, 2013. Association for Computational Linguistics.

[4] Z. E. Alexandrova. *Dictionary of Russian Synonyms*. Moscow: Russkii Yazyk, 2001.

[5] Y. Bengio, R. Ducharme, and P. Vincent. A neural probabilistic language model. *Journal of Machine Learning Research*, 3:1137–1155, 2003.

[6] Y. Bengio, H. Schwenk, J.-S. Senécal, F. Morin, and J.-L. Gauvain. Neural probabilistic language models. In *Innovations in Machine Learning*, pages 137–186. Springer, 2006.

[7] J. Bergstra, O. Breuleux, F. Bastien, P. Lamblin, R. Pascanu, G. Desjardins, J. Turian, D. Warde-Farley, and Y. Bengio. Theano: a CPU and GPU math expression compiler. In *Proceedings of the Python for Scientific Computing Conference (SciPy)*, 2010. Oral Presentation.

[8] J. A. Botha and P. Blunsom. Compositional morphology for word representations and language modelling. In *Proceedings of the 31th International Conference on Machine Learning, ICML 2014, Beijing, China, 21-26 June 2014*, pages 1899–1907, 2014.

[9] P. F. Brown, P. V. deSouza, R. L. Mercer, V. J. D. Pietra, and J. C. Lai. Class-based n-gram models of natural language. *Comput. Linguist.*, 18(4):467–479, 1992.

[10] S. F. Chen and J. Goodman. An empirical study of smoothing techniques for language modeling. In *Proceedings of the 34th Annual Meeting on Association for Computational Linguistics*, ACL '96, pages 310–318, Stroudsburg, PA, USA, 1996. Association for Computational Linguistics.

[11] F. Chollet. Keras. https://github.com/fchollet/keras, 2015.

[12] J. Chung, K. Cho, and Y. Bengio. A character-level decoder without explicit segmentation for neural machine translation. *CoRR*, abs/1603.06147, 2016.

[13] R. Cotterell, H. Schütze, and J. Eisner. Morphological smoothing and extrapolation of word embeddings. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*, 2016.

[14] S. Dieleman, J. Schlüter, C. Raffel, E. Olson, S. K. Sønderby, D. Nouri, D. Maturana, M. Thoma, E. Battenberg, J. Kelly, J. D. Fauw, M. Heilman, D. M. de Almeida, B. McFee, H. Weideman, G. Takács, P. de Rivaz, J. Crall, G. Sanders, K. Rasul, C. Liu, G. French, and J. Degrave. Lasagne: First release, 2015.

[15] Y. Goldberg. A primer on neural network models for natural language processing. *CoRR*, abs/1510.00726, 2015.

[16] J. T. Goodman. A bit of progress in language modeling. *Comput. Speech Lang.*, 15(4):403–434, 2001.

[17] A. Graves, S. Fernández, and J. Schmidhuber. Bidirectional LSTM networks for improved phoneme classification and recognition. In *Artificial Neural Networks: Formal Models and Their Applications - ICANN 2005, 15th International Conference, Warsaw, Poland, September 11-15, 2005, Proceedings, Part II*, pages 799–804, 2005.

[18] A. Graves and J. Schmidhuber. Framewise phoneme classification with bidirectional LSTM and other neural network architectures. *Neural Networks*, 18(5-6):602–610, 2005.

[19] K. Kann and H. Schütze. Single-model encoder-decoder with explicit morphological representation for reinflection. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 2: Short Papers*, 2016.

[20] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014.

[21] R. Kneser and H. Ney. Improved backing-off for m-gram language modeling. In *Acoustics, Speech, and Signal Processing, 1995. ICASSP-95., 1995 International Conference on*, volume 1, pages 181–184 vol.1, 1995.

[22] M. Korobov. Morphological analyzer pymorphy. http://pymorphy.readthedocs.io/.

[23] O. Kozlowa and A. Kutuzov. Improving distributional semantic models using anaphora resolution during linguistic preprocessing. In *Proceedings of International Conference on Computational Linguistics "Dialogue 2016"*, 2016.

[24] Y. LeCun, K. Kavukcuoglu, and C. Farabet. Convolutional networks and applications in vision. In *International Symposium on Circuits and Systems (ISCAS 2010), May 30 - June 2, 2010, Paris, France*, pages 253–256, 2010.

[25] W. Ling, C. Dyer, A. W. Black, I. Trancoso, R. Fermandez, S. Amir, L. Marujo, and T. Luis. Finding function in form: Compositional character models for open vocabulary word representation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1520–1530, Lisbon, Portugal, 2015. Association for Computational Linguistics.

[26] M.-T. Luong, R. Socher, and C. D. Manning. Better word representations with recursive neural networks for morphology. In *CoNLL*, Sofia, Bulgaria, 2013.

[27] T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781, 2013.

[28] T. Mikolov, M. Karafiát, L. Burget, J. Cernockỳ, and S. Khudanpur. Recurrent neural network based language model. *INTERSPEECH*, 2:3, 2010.

[29] T. Mikolov, S. Kombrink, L. Burget, J. H. Černockỳ, and S. Khudanpur. Extensions of recurrent neural network language model. In *Acoustics,*

Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on, pages 5528–5531. IEEE, 2011.

[30] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. *CoRR*, abs/1310.4546, 2013.

[31] A. Mnih and G. E. Hinton. A scalable hierarchical distributed language model. In *Advances in neural information processing systems*, pages 1081–1088, 2009.

[32] A. Panchenko, N. Loukachevitch, D. Ustalov, D. Paperno, C. M. Meyer, and N. Konstantinova. Russe: The first workshop on russian semantic similarity. In *Proceedings of the International Conference on Computational Linguistics and Intellectual Technologies (Dialogue)*, pages 89–105, 2015.

[33] J. Pennington, R. Socher, and C. Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar, 2014. Association for Computational Linguistics.

[34] M. Ranzato, G. E. Hinton, and Y. LeCun. Guest editorial: Deep learning. *International Journal of Computer Vision*, 113(1):1–2, 2015.

[35] R. Soricut and F. Och. Unsupervised morphology induction using word embeddings. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1627–1637, Denver, Colorado, 2015. Association for Computational Linguistics.

[36] X. Zhang, J. Zhao, and Y. LeCun. Character-level convolutional networks for text classification. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 649–657. Curran Associates, Inc., 2015.