Applied Computer Systems
ISSN 2255-8691 (online)
ISSN 2255-8683 (print)
December 2021, vol. 26, no. 2, pp. 132–138
https://doi.org/10.2478/acss-2021-0016
https://content.sciendo.com

Evaluation of Word Embedding Models in Latvian NLP Tasks Based on Publicly Available Corpora

Rolands Laucis[1*], Gints Jēkabsons[2]
[1,2] *Riga Technical University, Riga, Latvia*

***Abstract* – Nowadays, natural language processing (NLP) is increasingly relaying on pre-trained word embeddings for use in various tasks. However, there is little research devoted to Latvian – a language that is much more morphologically complex than English. In this study, several experiments were carried out in three NLP tasks on four different methods of creating word embeddings: word2vec, fastText, Structured Skip-Gram and ngram2vec. The obtained results can serve as a baseline for future research on the Latvian language in NLP. The main conclusions are the following: First, in the part-of-speech task, using a training corpus 46 times smaller than in a previous study, the accuracy was 91.4 % (versus 98.3 % in the previous study). Second, fastText demonstrated the overall best effectiveness. Third, the best results for all methods were observed for embeddings with a dimension size of 200. Finally, word lemmatization generally did not improve results.**

***Keywords* – Named entity recognition, natural language processing, part-of-speech tagging, word analogy, word embeddings.**

I. Introduction

It has been shown that using dense, continuous real-vector space word representations – word embeddings – yields significantly better results in natural language processing (NLP) tasks than older methods [1], [2]. There are various techniques for building word embeddings, such as building a statistical global co-occurrence matrix, probabilistic models, or using neural networks. Using these techniques, many methods and models have been developed and each of them, in their original papers, demonstrates better results than others. However, an extensive study conducted in 2019 concluded that no particular method significantly outperformed others in all evaluations for English [3].

This study focuses on word embeddings for the Latvian language, which has much richer morphology compared to English and has much smaller corpora readily available for training embeddings. This poses additional challenges in creating word embeddings for Latvian.

A few studies have been performed with regard to embeddings in some Latvian NLP tasks [4]–[6]; however, their experimental scope has limitations and the embeddings are created using corpora that are not publicly available.

The present study evaluates and compares embeddings inferred by a variety of methods using publicly available corpora, namely Latvian Wikipedia and the EuroParl [7] corpus. The methods are evaluated on the basis of three NLP tasks: word analogy, part-of-speech tagging (POS tagging) and named entity recognition (NER). The evaluation is performed by varying embedding dimensionality and experimenting with word lemmatization. As a result, a range of baseline results is obtained providing new insights on word embedding for Latvian.

The rest of the paper is organised as follows: Section I reviews previous work. Section II briefly presents each method evaluated in this study. Section III describes the experimental setup. Section IV presents experimental results and their analysis. Section V concludes the paper.

II. Related Work

In 2016, a study on word embedding methods specifically focusing on the Latvian language was conducted [4]. It compared five embedding models – Continuous Bag of Words (CBOW), Skip-Gram (SG), Structured Skip-Gram (SSG), Continuous Window (CWIN) and Character-based word embeddings (CWE) with dimension sizes of 20, 50, 100 and 200. Three NLP tasks were used to evaluate the embeddings, achieving state-of-the-art results in Latvian POS tagging (98.3 % accuracy) and Latvian NER (90.2 % accuracy) tasks. The study found that SSG (100 and 200 dimensions), CWIN (100 dimensions), and SSG (100 dimensions) concatenated with lemmatized SG (100 dimensions) produced the best results. Lemmatization alone did not improve results in these tasks.

More recently, there have been two NLP studies on the Latvian language using "Bidirectional Encoder Representations from Transformers" (BERT) method [8], which is slightly different from word embeddings, since it learns the embeddings and solves arbitrary downstream NLP tasks at the same time, and only needs fine-tuning for specific tasks.

The LVBERT model [5] yields slightly worse results for POS tagging (98.1% accuracy) and NER (82.6 % accuracy) tasks than the best embeddings in [4]. Likewise, the Latvian BERT model in [6]applied to NER task achieved an F_1 score of

[*]Corresponding author's e-mail: rolands.laucis@gmail.com

81.91 % while using a significantly smaller training corpus than in [4].

In addition, a 2021 study compares BERT and ELMo ("Embeddings from Language Models" [9]) models with a fastText baseline on multiple NLP tasks for multiple European languages, including Latvian [10]. Their fastText model results show 55.7 % F_1 score in NER task and 46.2 % accuracy in UPOS task.

III. Word Embedding Models

In 2019, an extensive study was conducted on embedding models for English [3]. The study evaluated word embeddings on five types of intrinsic evaluations and five types of extrinsic evaluations. It was concluded that no particular method significantly outperformed others in all evaluations for English. In addition, the SG model was found to be the least volatile across all evaluation results. Observing the results in [3], a set of four methods was chosen for experiments in this paper. These methods are listed and shortly described in the next four paragraphs.

Word2vec [1] introduces two neural network architectures – CBOW and SG – for creating word embeddings that outperformed state-of-the-art deep learning models with a large unannotated corpus in analogy and word similarity tasks [1], [11]. Iterating over the corpus tokens, CBOW is tasked with predicting the current token from a window of surrounding tokens, whereas SG predicts surrounding tokens from the current token. Overall, both architectures seem to have similar performance, but SG is less volatile [1], [3], [12].

SSG method [12] alters word2vec Skip-Gram's objective function to account for the distance of window tokens achieving better results in syntax-centred tasks like POS tagging.

FastText [13] improves SG by introducing sub-word vector representations, the sum of which is equal to the word vector. This is useful for languages with many word inflections as well as working with out-of-vocabulary words.

Ngram2vec [14] adapts SG to work with token n-grams, thus putting more emphasis on word combinations that can be found in natural language, which arguably can generate better embeddings.

IV. Experimental Setup

A. Corpus for Training Embedding Models

Firstly, raw text was gathered from Latvian Wikipedia and the publicly available EuroParl Latvian corpus [7], being a large collection of the European Parliament proceedings from 1996 to 2011. The text was concatenated and then processed with the following steps, creating a cleaned corpus (hereafter referred to as the regular corpus):

- all abbreviations in the form of two or more capitalized letter sequences were removed;
- case folding – all letters were lowercased;
- all character sequences of up to 100 characters long between brackets were removed. This is because in Latvian Wikipedia these mostly contain word translations into foreign languages while in EuroParl these contain repetitive annotations;
- periods and commas followed by a space were separated from a preceding word or number by a space;
- all characters, except alpha-numeric characters, periods, commas, carriage returns, line feeds and spaces, were removed;
- the corpus could then be tokenized by splitting on whitespace.

After preparing the regular corpus, a lemmatized version of the same corpus was prepared by replacing all words with their lemmas. Lemmatization is done with the UDPipe 1 [15] tool using Latvian UDPipe 2.5 model. This tool has won a few worldwide competitions and the Latvian model has achieved 92.7 % lemmatization accuracy [15]. Statistics of created corpora and their comparison with corpora from previous works is shown in Table I. Notably, the created corpus is about 46 times smaller by token count than the corpus in [4] or [6].

TABLE I
Statistics of Created Corpora and their Comparison to Corpora in Previous Works

	Vocabulary size		**Token count**
	All	**Min freq 10**	
Regular corpus	877 242	130 976	37 035 858
Lemmatized corpus	597 746	78 812	
Znotiņš [4]	-	968 000	1 700 000 000
Znotiņš lemmatized [4]	-	547 000	
Vīksna & Skadiņa [6]	-	-	1 600 000 000
Znotiņš & Barzdiņš [5]	-	-	500 000 000

B. Embedding Training Parameters

Word embeddings are trained on the regular corpus and the lemmatized corpus with varying dimension sizes – 50, 100, 200, and 300. Word2vec and fastText are trained using the Python Gensim library [16]. Ngram2vec [17] and SSG [18] are trained using the original software tools provided by their authors. All the tools are used with their default parameters with few exceptions: the training on a corpus is done with 12 epochs and the window size is set to 5, as it is the default for a few of these tools and previous works also use this value. Ngram2vec is configured to produce bi-grams. All methods are set to the Skip-Gram architecture.

C. Evaluation Tasks and Datasets

As stated above, the embedding models are evaluated on the basis of three NLP tasks: word analogy, part-of-speech tagging (POS tagging) and named entity recognition (NER).

Word analogy task is a direct intrinsic evaluation of word embeddings [3]. It is possible to find a word that is analogous to a given set of words, for example, "king is to queen as man is to woman", where the word "woman" can be found by using the word vectors in a mathematical equation, e.g., *woman* = *man* – *king* + *queen* [2].

Since a native Latvian analogy dataset is not available, a dataset created in [19] for Slovenian, that was translated to many European languages, including Latvian, was used. It has a total of 20 138 analogies across 15 categories and is publicly available [20].

POS tagging is a popular task for extrinsic evaluation of word embeddings. The task is to tag morphological features for each token in a document either as coarse-grained POS (tagging just part of speech – "UPOS") or fine-grained POS (tagging full morphological information – "XPOS"). In the present study, this task is done by training a neural network model with Python SpaCy [21] using components *tagger* and *morphologizer* with default settings (choosing the "accuracy" preset from the available ones), except epochs that are set to 12. The "Latvian TreeBank" dataset [22] is used for training and evaluating each model. It has a total of 13 643 sentences and 219 955 tokens.

NER is also a popular task for extrinsic evaluation of word embeddings. The task is to tag each token of a document with a classification from a set of entity types, e.g., *event*, *location*, etc. In the present study, this task is done by training a neural network model with Python SpaCy using component *ner* with the same settings as for POS task. The "LUMII-AILab" dataset [23] is used for training and evaluating each model. It has a total of 7476 sentences that are split into training, development and test subsets in percentage proportions 60-30-10, resulting in subsets of 4486, 2243 and 747 sentences, respectively.

An overview of the evaluation datasets used in this paper is shown in Table II.

TABLE II

OVERVIEW OF WORD EMBEDDING EVALUATION TASK DATASET

Analogy dataset		
Analogy count	**Category count**	**Source**
20138	15	[19] translated dataset
POS tagging dataset		
Sentence count	**Category count (Universal POS tags)**	**Source**
13643	17	Latvian TreeBank [22]
NER dataset		
Sentence count	**Number of entity types**	**Source**
7476	9	LUMII-AILab [23]

V. RESULTS

A. Analogy Task Results

Analogy evaluations are measured as percent answered correctly out of all analogies that the model could theoretically answer using the *3CosMul* method [24], per analogy category and overall. The number of analogies that a model can answer is the analogy count where all four words per analogy have a vector within the model. A correctly answered analogy is counted in a "top 1" setting where the answer is correct only if the closest vector in the embeddings space corresponds to the correct word. The overall results for all methods can be seen in Table III. More detailed results by analogy categoryfor dimension size 200are given in Table IV and Table V. It is clear that fastText has the best results overall. This is likely due to fastText method of representing words, which is useful for word forms in morphologically rich languages. Lemmatization helps improve performance on word2vec and SSG models but, as expected, quite a few categories show 0 % answered. This means that lemmatization is able to improve results if these few categories are not important for a given practical application. It is also apparent that non-lemmatized fastText consistently gets the best results in word form manipulation categories, as can be expected. Yet lemmatized SSG consistently gives the best results in the "family" and "city-in-country" categories, but ngram2vec does not outperform other models in any category.

It should be noted that the results are percentage from all theoretically answerable analogies, which might be a different count for each method. Since these results reflect the performance of each model only for the analogies with words that it contains and not for the whole set of analogies, counts of theoretically answerable analogies are gathered in Table VI and a normalization formula using this table is introduced: overall result from Table III, multiplied by theoretically answerable analogy count from Table VI, divided by total analogy count from the dataset. Results using the normalization formula are given in Table VII and Fig. 1.

TABLE III

ANALOGY TASK RESULTS AS PERCENTAGE ANSWERED CORRECTLY FOR ALL MODELS TRAINED WITH REGULAR AND LEMMATIZED CORPUS BY DIMENSIONALITY

Model		**50 d.**	**100 d.**	**200 d.**	**300 d.**
fastText	Regular	**12.36**	**20.96**	**24.66**	**23.39**
	Lemmatized	10.62	15.38	16.22	14.73
word2vec	Regular	5.23	9.00	10.20	9.44
	Lemmatized	8.07	12.99	14.86	14.76
SSG	Regular	6.58	8.60	9.45	8.67
	Lemmatized	9.38	15.17	16.90	14.83
ngram2vec	Regular	1.47	1.54	1.56	1.69
	Lemmatized	0.75	0.97	0.98	0.99

TABLE IV

ANALOGY TASK DETAILED RESULTS AS PERCENTAGE ANSWERED CORRECTLY FOR ALL MODELS TRAINED WITH REGULAR CORPUS

Analogy category	**fastText**	**word2vec**	**SSG**	**ngram2vec**
Overall	**24.66**	10.20	9.45	1.56
capitals-countries	**10.90**	8.65	6.09	1.59
family	22.33	26.28	**29.21**	9.74
city-in-country	**16.40**	15.00	11.70	0.76
animals	0.07	0.82	**1.50**	0.00
city-river	0.00	0.00	0.00	0.00
gram1-adjective-to-adverb	**50.86**	2.80	3.33	0.22
gram2-opposite	**80.71**	0.48	1.67	0.00
gram3-comparative	**82.54**	31.61	29.77	3.56
gram4-superlative	**21.17**	2.83	5.48	0.50
gram5-verbal-noun	**66.01**	3.36	6.43	0.00
gram6-nationality	**56.21**	9.08	8.50	3.04
gram7-plural	**26.47**	12.82	10.66	2.56
gram8-genitive-dative	**29.24**	5.04	6.82	0.38
gram9-present-past	**38.95**	27.36	29.53	5.43
gram0-present-future	16.60	14.23	**26.58**	0.99

TABLE V

Analogy Task Detailed Results as Percentage Answered Correctly for All Models Trained with Lemmatized Corpus

Analogy category	fastText	word2vec	SSG	ngram2vec
Overall	16.22	14.86	**16.90**	0.98
capitals-countries	9.50	17.80	**16.64**	1.29
family	25.54	31.17	**45.32**	4.33
city-in-country	12.59	22.78	**23.25**	0.55
animals	0.14	**7.18**	6.46	0.28
city-river	0.51	3.89	**6.43**	1.52
gram1-adjective-to-adverb	**45.05**	10.86	21.67	0.65
gram2-opposite	**65.42**	7.91	12.11	0.79
gram3-comparative	**26.67**	0.00	6.67	0.00
gram4-superlative	0.00	0.00	0.00	0.00
gram5-verbal-noun	**38.93**	7.71	9.31	0.79
gram6-nationality	**28.43**	1.96	6.94	0.37
gram7-plural	**12.87**	0.00	0.00	0.00
gram8-genitive-dative	0.00	0.00	0.00	0.00
gram9-present-past	0.00	0.00	0.00	0.00
gram0-present-future	0.00	0.00	0.00	0.00

TABLE VI

Counts of Theoretically Answerable Analogies and Percentage from Total Analogies in the Dataset

Model		Analogy count	Percent answer
fastText	Regular	18 794	93.33
	Lemmatized	12 260	60.88
word2vec	Regular	18 794	93.33
	Lemmatized	12 260	60.88
SSG	Regular	16 588	82.37
	Lemmatized	10 062	49.97
ngram2vec	Regular	18 156	90.16
	Lemmatized	12 166	60.41
Total analogy count		20 138	-

TABLE VII

Analogy Task Normalized Results as Percentage Answered Correctly for All Models Trained with Regular and Lemmatized Corpus by Dimensionality

Model		50 d.	100 d.	200 d.	300 d.
fastText	Regular	**11.54**	**19.56**	**23.02**	**21.83**
	Lemmatized	6.47	9.36	9.88	8.97
word2vec	Regular	4.88	8.40	9.52	8.81
	Lemmatized	4.91	7.91	9.05	8.98
SSG	Regular	5.42	7.08	7.78	7.14
	Lemmatized	4.69	7.58	8.44	7.41
ngram2vec	Regular	1.32	1.39	1.41	1.52
	Lemmatized	0.45	0.59	0.59	0.60

As expected, fastText and word2vec have the same number of analogies while ngram2vec has a different count because it learns from pairs of words.

It is again apparent that the fastText non-lemmatized model significantly outperforms all other models and a dimension size of 200 yields the best results for all models, so one can conclude that the optimal dimension size is somewhere between 100 and 300.

No other studies were found for the Latvian language that evaluated embedding models with the analogy task, so the fastText model of this study was compared to the publicly available fastText Latvian model made by the original authors of fastText [13] (hereinafter referred to as the large fastText model). The detailed results are shown in Table VIII and overall normalized results are shown in Table IX.

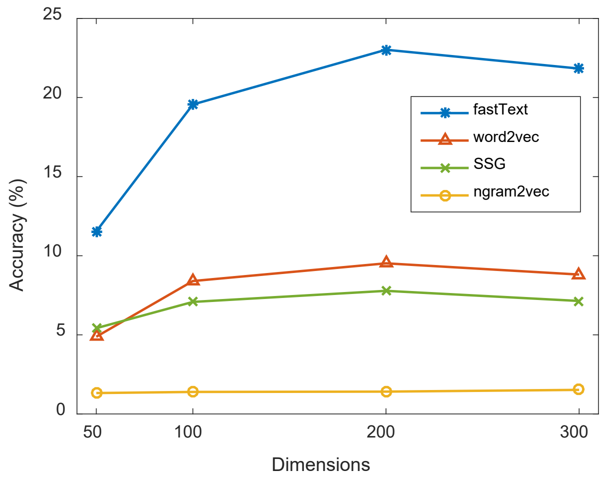

Fig. 1. Analogy task normalized results (top 1 answer) for non-lemmatized models by a number of dimensions for each method.

TABLE VIII

Analogy Task Results for this Paper fastText 300 Dimension Model in Comparison with the Large fastText Model across All Categories

Analogy category	fastText		Large fastText
	Regular	Lemmatized	
Overall	23.39	14.73	**32.11**
capitals-countries	8.00	8.03	**26.21**
family	19.57	16.45	**40.71**
city-in-country	11.51	10.51	**35.52**
animals	0.00	**0.21**	**0.21**
city-river	0.00	0.17	**1.56**
gram1-adjective-to-adverb	**53.01**	41.61	38.71
gram2-opposite	**82.62**	67.79	63.95
gram3-comparative	**85.05**	34.29	67.86
gram4-superlative	**17.67**	0.00	10.45
gram5-verbal-noun	**69.76**	35.97	52.90
gram6-nationality	**61.72**	27.12	29.31
gram7-plural	26.03	12.50	**32.69**
gram8-genitive-dative	30.25	0.00	**36.19**
gram9-present-past	33.70	3.57	**71.20**
gram0-present-future	18.18	0.00	**56.16**

Their model performs about 42 % or 9 percentage points better than the non-lemmatized model of this study, which is expected because, although both do use Latvian Wikipedia in their training corpus, large fastText also uses Common Crawl [25] Latvian corpus which is much larger and contains texts on a wider range of topics. Interestingly, the fastText model of this study is still better than the other models on the same categories as before and so the large fastText model only outperforms in other categories.

TABLE IX

ANALOGY TASK NORMALIZED OVERALL RESULTS FOR THIS PAPER FASTTEXT 300 DIMENSION MODEL IN COMPARISON WITH THE LARGE FASTTEXT MODEL

fastText		Large fastText
Regular	**Lemmatized**	
21.83	8.97	31.05

B. Part-of-Speech Task Results

The POS tagging task has two results – UPOS (coarse-grained) prediction accuracy and XPOS (fine-grained) prediction accuracy. UPOS results for all methods and dimension sizes are shown in Fig. 2 and XPOS in Fig. 3. As expected from the previous results, fastText gives the best results for UPOS and XPOS tagging. Additionally, for XPOS there is a bigger gap between fastText and other models. However, in both evaluations SSG slightly outperforms word2vec contrary to the analogy task. Chiefly, ngram2vec retains similar result patterns as in the analogy task. Overall, unlike in the analogy task, the dimensionality of 300 seems to be still converging to the best results for all models, but the increase is small.

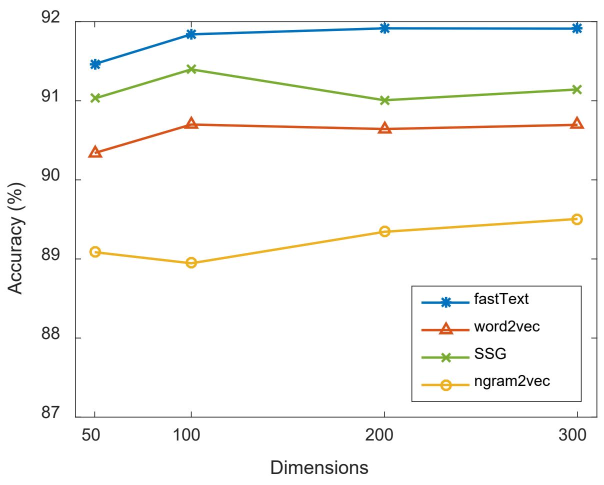

Fig. 2. UPOS accuracy: percentage answered correctly for all non-lemmatized models and dimension sizes.

A comparison is made with previous works for UPOS accuracy (Table X). Since there are few previous works to compare, the study by Paikens [26] is also included, having results of a UPOS tagging task for the Latvian language using deep neural networks without separately prepared word embeddings. As can be seen in Table X, the previous works have impressive results and differ in only few tenths of a percentage point, whereas the SSG models created in this study yield results that are about 7 percentage points worse. Nevertheless, these results are notably impressive as well, given that 46 times smaller corpus was used.

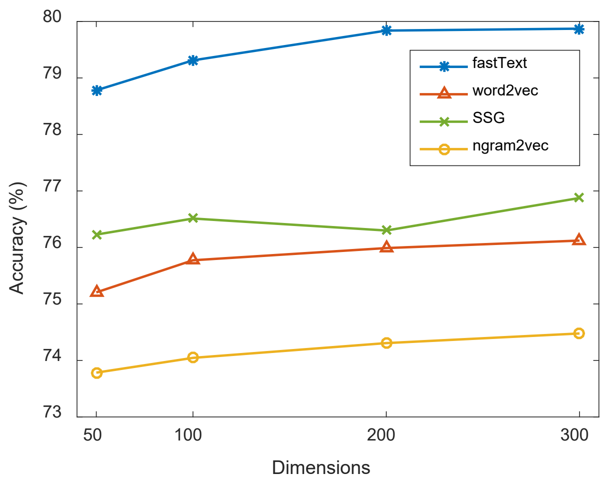

Fig. 3. XPOS accuracy: percentage answered correctly for all non-lemmatized models and dimension sizes.

TABLE X

UPOS TASK ACCURACY RESULT COMPARISON WITH PREVIOUS WORKS SSG WORD EMBEDDINGS AND CUSTOM NEURAL NETWORKS

	Corpus size (in millions of tokens)	**SSG-100, %**	**SSG-200, %**	**NN, %**
This paper	37	91.40	91.01	–
Znotiņš [4]	1700	98.30	98.30	–
Paikens [26]	–	–	–	97.8
Vīksna & Skadiņa [6]	1600	–	–	98.1

C. Named Entity Recognition Task Results

The used NER dataset has nine entity categories: *organisation*, *money*, *location*, *time*, *product*, *event*, *entity*, *person* and *geopolitical entity*. F_1 score results for the NER task are shown in Fig. 4. The results are varied between models and inconsistent within models across dimension sizes. In comparison with the POS task results, there is no clear superior model in this evaluation. The worst performing model, on average, is fastText. Word2vec mildly follows ngram2vec volatility and has the best result of all at 59.4 % with 300 dimensions.

Comparing NER results of SSG 100- and 200-dimension word embeddings with previous works in Table XI, it is clear that these models significantly underperform by an average of 30 percentage points. 46 times smaller training corpus could explain this difference; furthermore, a closer look at the trained NER model revealed that about 19 % of NER dataset words were not represented in the embeddings.

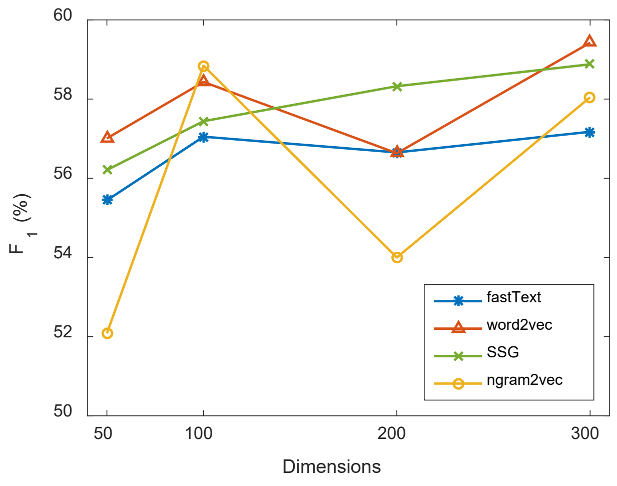

Fig. 4. NER task F_1 score results for all non-lemmatized models and dimensions.

TABLE XI

NER TASK F_1SCORE RESULT COMPARISON WITH PREVIOUS WORKS (SSG WORD EMBEDDINGS AND CUSTOM NEURAL NETWORKS)

	Corpus size (millions of tokens)	**SSG-100, %**	**SSG-200, %**	**NN, %**
This paper	37	57.44	58.32	–
Znotiņš [4]	1700	89.00	89.10	–
Vīksna & Skadiņa [6]	1600	–	–	82.60
Znotiņš & Barzdiņš [5]	500	–	–	79.72

VI. CONCLUSION

Numerous observations and conclusions can be made from the results of this study. Firstly, in the UPOS task, using a training corpus that is 46 times smaller than in a previous study, the accuracy achieved was 91.4 % (versus 98.3 % in the previous study). Secondly, the fastText method achieved the best results compared to the other methods in this study, whereas the ngram2vec showed the worst performance. Thirdly, the best results for all methods were observed for embeddings with a dimension size of 200, and the most effective dimension size was between 100 and 300. Fourthly, it was found that lemmatization generally did not improve results for any method with a few exceptions: an improvement was achieved on word2vec and SSG models when performance was measured on analogy subtasks where inflections were not necessary: *capitals-countries*, *family*, *city-in-country*, *animals*, and *city-river*.

As a possible future research area, it would be interesting to study the impact of window size and other hyperparameters on the results in analogy, POS, and NER tasks, as well as to broaden the set of tasks.

REFERENCES

[1] T. Mikolov, K. Chen, G. Corrado, and J. Dean, “Efficient estimation of word representations in vector space,” in *Workshop Track Proceedings of 1st International Conference on Learning Representations*, Scottsdale, Arizona, USA, May 2013, pp. 1–12.

[2] P. Jeffrey, S. Richard, and D. M. Christopher, “GloVe: Global vectors for word representation,” in *Empirical Methods in Natural Language Processing (EMNLP)*, 2014, pp. 1532–1543.

[3] B. Wang, A. Wang, F. Chen, Y. Wang, and C.-C. J. Kuo, “Evaluating word embedding models: methods and experimental results,”*APSIPA Transactions on Signal and Information Processing*, vol. 8, Art no. e19, Jul. 2019. https://doi.org/10.1017/ATSIP.2019.12

[4] A. Znotiņš, “Word embeddings for Latvian natural language processing tools,”*Human Language Technologies – The Baltic Perspective*, vol. 289, IOS Press, pp. 167–173, 2016. https://doi.org/10.3233/978-1-61499-701-6-167

[5] A. Znotiņš and G. Barzdiņš, “LVBERT: Transformer-based model for Latvian language understanding,” *Human Language Technologies – The Baltic Perspective*, vol. 328, IOS Press, pp. 111–115, 2020. https://doi.org/10.3233/FAIA200610

[6] R. Vīksna and I. Skadiņa, “Large language models for Latvian named entity recognition,” *Human Language Technologies – The Baltic Perspective*, vol. 328,IOS Press, pp. 62–69, 2020. https://doi.org/10.3233/FAIA200603

[7] “EuroParl,” [Online]. Available: https://www.statmt.org/europarl/. Accessed on: May 2021.

[8] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “BERT: Pre-training of deep bidirectional transformers for language understanding”, in *Proceedings of NAACL-HLT*, Minneapolis, Minnesota, 2019, p. 4171–4186.

[9] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, “Deep contextualized word representations,” *arXiv*, Art no.1802.05365, pp. 1–15, 2018.

[10] M. Ulčar, A. Žagar, C. Armendariz, A. Repar, S. Pollak, M. Purver, and M. Robnik-Šikonja, “Evaluation of contextual embeddings on less-resourced languages,” *arXiv*, Art no. 2107.10614, pp. 1–45, 2021.

[11] X. Rong, “word2vec parameter learning explained”, *arXiv*, Art no. 1411.2738v4, pp. 1–21, 2016.

[12] W. Ling, C. Dyer, A. Black, and I. Trancoso, “Two/Too simple adaptations of Word2Vec for syntax problems”, in *Proceedings of the 2015 Conference of the North American*, Denver, Colorado, May-June 2015, pp. 1299–1304. https://doi.org/10.3115/v1/N15-1142

[13] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, “Enriching word vectors with subword information”, *Transactions of the Association for Computational Linguistics*, vol. 5, June 2017, pp. 135–146. https://doi.org/10.1162/tacl_a_00051

[14] Z. Zhao, T. Liu, S. Li, and B. Li, “Ngram2vec: Learning improved word representations from Ngram co-occurrence statistics”, in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, Copenhagen, Denmark, Sep. 2017, pp. 244–253. https://doi.org/10.18653/v1/D17-1023

[15] “UDPipe,” [Online]. Available: https://ufal.mff.cuni.cz/udpipe/1. Accessed on: May 2021.

[16] “Gensim library,” [Online]. Available: https://radimrehurek.com/gensim/. Accessed on: May 2021.

[17] “Ngram2vec tool repository,” [Online]. Available: https://github.com/zhezhaoa/ngram2vec. Accessed on: May 2021.

[18] “Structured Skip-Gram tool repository,” [Online]. Available: https://github.com/wlin12/wang2vec. Accessed on: May 2021.

[19] M. Ulčar, K. Vaik, J. Lindstrom, M. Dailidenaite, and M. Robnik-Sikonja, “Multilingual culture-independent word analogy datasets,” in *Proceedings of the 12th Language Resources and Evaluation Conference, LREC 2020*, Marseille, France, May 2020, pp. 4074–4080.

[20] “Translated analogy dataset repository,” [Online]. Available: https://www.clarin.si/repository/xmlui/handle/11356/1261. Accessed on: May 2021.

[21] “SpaCy tool,” [Online]. Available: https://spacy.io/. Accessed on: May 2021.

[22] “LVTB dataset repository.” [Online]. Available: https://github.com/UniversalDependencies/UD_Latvian-LVTB/tree/master. Accessed on: May 2021.

[23] “LUMII_AiLab NER dataset repository.” [Online]. Available: https://github.com/LUMII-AILab/FullStack/tree/master/NamedEntities. Accessed on: May 2021.

[24] O. Levy and Y. Goldberg, "Linguistic Regularities in Sparse and Explicit Word Representations", in *Proceedings of the Eighteenth Conference on Computational Natural Language Learning*, June 2014, pp. 171–180. https://doi.org/10.3115/v1/W14-1618

[25] "CommonCrawl," [Online]. Available: https://commoncrawl.org/. Accessed on: May 2021.

[26] P. Paikens, "Deep neural learning approaches for Latvian morphological tagging," *Human Language Technologies – The Baltic Perspective*, vol. 289,IOS Press, pp. 160–166, 2016.https://doi.org/10.3233/978-1-61499-701-6-160

Rolands Laucis received his Bachelor degree from Riga Technical University in 2021. At present, he is a Research Assistant at the Institute of Applied Computer Systems of Riga Technical University. His current research interests include natural language processing and machine learning.
E-mail: rolands.laucis@gmail.com
Personal webpage: https://rolandslaucis.lv

Gints Jēkabsons received his Doctoral degree from Riga Technical University in 2009. At present, he is an Associate Professor and Researcher at the Department of Software Engineering and Institute of Applied Computer Systems of Riga Technical University. His current research interests include information retrieval, machine learning and natural language processing.
E-mail: gints.jekabsons@rtu.lv
ORCID iD: https://orcid.org/0000-0002-9575-2488