Intrinsic word embedding model evaluation for Lithuanian language using adapted similarity and relatedness benchmark datasets

Mindaugas Petkevičius[a], Daiva Vitkutė-Adžgauskienė[a]

[a] *Vytautas Magnus University, K. Donelaičio g. 58, Kaunas, 44248, Lithuania*

Abstract

Word embeddings are real-valued word representations capable of capturing lexical semantics and trained on natural language corpora. Word embedding models have gained popularity in recent years, but the issue of selecting the most adequate word embedding evaluation methods remains open. This paper presents research on adaptation of the intrinsic similarity and relatedness task for the Lithuanian language and the evaluation of word embedding models, testing the quality of representations independently of specific natural language processing tasks. 7 different evaluation benchmarks were adapted for the Lithuanian language and 50 word embedding models were trained using *fastText*, *GloVe,* and *Word2vec* algorithms and evaluated on syntactic and semantic similarity tasks. The obtained results suggest that for the intrinsic similarity and relatedness task, the dimension parameter has a significant impact on the evaluation results, with larger word embedding dimension yielding better results.

Keywords

Word embeddings, evaluation, Lithuanian language, word2vec, fastText, GloVe

1. Introduction

The development of natural language processing tools influenced a growing need for word embeddings as real-valued representations of words for text analytics, generated by applying distributive semantic models. While word embeddings have become one of the most widely used tools in modern natural language processing (NLP) applications, their limitations have not yet been fully explored. The problem of assessing word embedding consistency and quality is one of the most relevant questions in distributive semantics research.

The idea of word embeddings is not new, but it gained popularity after Mikolov et al. [1] presented the *Word2vec* model in 2013. The *fastText* model, developed by Facebook AI Research (FAIR), introduces embeddings using subword information. The next big improvement came from Stanford with *GLoVE* (Global-Vectors) [2], based on word-word co-occurrence statistics in a corpus.

There are two types of word embedding evaluation: intrinsic and extrinsic. Intrinsic evaluation tests the representation quality independent of specific natural language processing (NLP) tasks, while extrinsic evaluation uses word embeddings as input features to an NLP task and measures changes in corresponding performance metrics. We focus on intrinsic evaluation methods, based on human annotated datasets, because datasets can be adapted for different languages by translating and reevaluating human annotated scores.

The method of word semantic similarity, based on correlation with human judgment of how closely words are related among themselves, was one of the first intrinsic evaluation metrics for distributional meaning representations. According to this method, words *smart* and *intelligent* should be closer in the vector space than *smart* and *dumb*, since *smart* and *intelligent* are intuitively better semantically related.

26th International Conference Information Society and University Studies - IVUS 2021
EMAIL: mindaugas.petkevicius@vdu.lt (A. 1); daiva.vitkute@vdu.lt (A. 2)
ORCID: 0000-0002-1120-4848 (A. 1); 0000-0001-7923-1087 (A. 2)

There are gold-standard benchmarks for evaluating distributive semantic models such as SimLex999 [3], MEN [4], etc., focused on semantic relatedness. These benchmarks consist of certain word pairs and their relative similarity scores. The similarity scores are defined in the interval between 0 and 10, e.g., the score for the words *book* and *paper* is *7.46*. When applied, these scores are compared with word pair *cosine* vector similarity results for word embeddings.

The word analogy method aims to identify words based on operation prediction in a word vector space. The method tries to predict a missing word in a word pair based on a known relationship in another word pair. Thus, for a dataset *a–b*, *c–d*, the task is to identify an unknown word *d* based on the known relationship between words *a* and *b*. For example, given the words, *a* (*brother*), *b* (*sister*), and *c* (*father*), this method should correctly predict the value *mother* for word *d* [5]. The Google analogy dataset [6] and BATS [7] are the most popular datasets. The Google test set has become the standard for word embedding analysis. BATS is a newer dataset that is much larger and more balanced.

The word clustering method evaluates a word embedding space by applying the word clustering approach. It is aimed at splitting a given word set into groups of words corresponding to different categories based on word vectors. For example, words *dog* and *cat* belong to one cluster, while words *car* and *plane* – to another [8].

The situation with word embeddings for the Lithuanian language is influenced by its specifics. The Lithuanian language is a morphologically rich Baltic language, being considered one of the most archaic living Indo-European languages [9]. It has a relatively large vocabulary, containing over 500 000 unique words [10]. On the other hand, the Lithuanian language lacks textual resources due to the small size of the nation using it. Lithuanian Wikipedia, for example, has 199 567 articles, while better represented languages have over a million each [11]. Several attempts were made to perform intrinsic and extrinsic evaluation of the Lithuanian language embeddings. However, so far there are no available semantic similarity benchmarks for this purpose.

The goal of this research was to adapt selected intrinsic similarity benchmarks for the Lithuanian language and to apply them for experimental evaluation of *fastText*, *Word2vec*, and *GloVe* embedding models with different hyperparameters.

In order to reach this goal, we perform the following tasks: related work analysis (Section 2), corpus building for embedding training (Section 3), methodology for the adaptation of evaluation benchmarks for the Lithuanian language (Section 4), experimental evaluation of different embeddings based on the derived benchmarks (Section 5), conclusions and future plans (Section 6).

2. Related Works

In recent years, there have been several critical articles on intrinsic assessment methods: some researchers address the subjectivity of human judgments, the vagueness of instructions for particular tasks, and terminology confusions [12]. However, despite these flaws, these methods are widely used for embedding model evaluation for different languages.

There have been successful attempts to adapt intrinsic evaluation benchmarks to other languages. Research has shown that when monolingual vector space models were translated into German, Russian, and Italian, it became clear that their predictions did not always correlate well with human decisions made in the language used for model training [13].

Another study attempted to translate the SymLex999 benchmark into Estonian and discovered that, unlike in the original research, computational word embedding models better correlate with noun scores rather than adjective scores [14].

A few studies on the evaluation of Lithuanian word embeddings have been carried out. In the first study, word embeddings for different models and training algorithms were evaluated against a limited implementation of the Lithuanian WordNet [15], showing that the Continuous Bag of Words (CBOW) approach performed significantly better than the skip-gram approach for *Word2vec* word embeddings, vector dimensions having little effect in this case.

The second study compared traditional and deep learning approaches for sentiment analysis using word embeddings, finding that deep learning performed well only when applied to small datasets, and that traditional methods performed better in all other contexts [16].

The third study was conducted with Transformer models using *GloVe* word embeddings [17]. The study concluded that multilingual transformer models can be fine-tuned to word vectors, but still perform much worse than specifically trained embeddings.

In conclusion, we see that the Lithuanian language lacks word embedding evaluation benchmarks.

3. Corpus

Semantic intrinsic similarity benchmarks cover many different types of test domains, such as geography, languages, currency, etc. Therefore, we need to have a wide variety of data for embedding training. Research has shown that for a larger corpus we get better word embeddings [18]. For this reason, it is important to build an extensive corpus for embedding training, that will be further used for evaluation.

Wikipedia texts are usually a typical approach for building a corpus for embedding training. In order to expand our experimental corpus, we used articles from Lithuanian news portals, mainly from the largest one, *Delfi.lt*, the collected articles covering different topical areas such as news, cars, fitness, culture, food, and so on.

In order to obtain better word embeddings, we also included texts from the Corpus of Contemporary Lithuanian Language (CCLL) [19], texts in a variety of genres and topics.

Statistics for our combined experimental corpus is presented in Table 1.

Table 1
Experimental corpus: initial version

Corpora	Documents	Token count	Unique tokens
Wikipedia	286 089	22 942 951	971 506
CCLL	8 128	136 279 087	2 329 976
News articles	118 930	60 456 637	718 051
Total	413 148	219 678 675	2 894 874

The pre-processing phase consists of two steps: 1) breaking text into tokens, lowercasing text, removing special symbols, numbers, non-Lithuanian words and stop-words 2) removing short documents, less than 50 characters in size; 3) lemmatizing the texts, this being important for rich morphology languages [20]. Lemmatization was performed using lexical and morphological analysis tools from the Lithuanian language technology infrastructure built in the Semantika2 [21] project.

Alternatively, text could have been stemmed instead, but lemmatization was preferred, as all our documents were in normative spelling and punctuation. Stemming is more favorable in case of social texts with lots of out-of-dictionary words. Also, stemming has its limitations, e.g. over-stemming and under-stemming problems [22].

The statistics for the final version of our experimental corpus are presented in Table 2.

Table 2
Experimental corpus: final version

Parameter	Value
Document count	303 443
Total tokens	160 174 732
Unique tokens	1 396 607

4. Methodology

The methodology part covers the methods applied in this research: (1) benchmark dataset adaptation method for semantic similarity based intrinsic embedding model evaluation; (2) semantic similarity based embedding evaluation using the adapted benchmarks.

4.1. Adaptation of benchmark datasets

As a result of a brief analysis, the following English-language benchmarks were selected for the adaptation to the Lithuanian language, their popularity being the main criteria:

1. MEN (Marco, Elia and Nam), 3 000 pairs [23].
2. WordSim-353, 353 pairs assessed by semantic similarity [24].
3. WordSim-353-REL, 252 pairs [25].
4. WordSim-353-SIM, 203 pairs [26].
5. SimLex-999, 999 pairs [27].
6. MTurk-287, 287 pairs [28].
7. RG-65, 65 pairs [29].

There are 5160-word pairs (2191 unique words) in total across all datasets.

The following algorithm was applied for the dataset adaptation to the Lithuanian language:

1. Automated translation of datasets (by applying the Google Cloud Translation API) [30].
2. Inconsistency checking (manual examination), discarding inconsistent word pairs.
3. Word lemmatization.
4. Re-evalution of the score that was initially assigned to the English language word pairs was done by two independent persons (manual procedure). An average score was calculated.

4.2. Embedding evaluation using semantic similarity benchmarks

As mentioned in Chapter 1, the semantic similarity datasets are based on correlation with human judgments of how closely words are related.

The similarity benchmark datasets consist of a certain number of word pairs. Each pair is determined by its similarity and relatedness. The values are in the range [0, 10], depending on the dataset.

The word embedding models are represented by corresponding vectors for each word in the dictionary. If a word is missing in the trained word embedding model, it is replaced by the mean of all vectors. In order to calculate the similarity between vectors, we can use the cosine similarity formula (see Eq. 1), where *a* and *b* are vectors in the word embedding vector space.

$$sim_{cos}\,(a_i, b_i) = \frac{\boldsymbol{a_i}\,.\,\boldsymbol{b_i}}{||\boldsymbol{a_i}|| \times\, ||\boldsymbol{b_i}||}, \tag{1}$$

here a_i and b_i are vectors of *N*-dimension. The result of cosine similarity is a value in the range [-1, 1] interval, where 1 stands for identical vectors, and -1 for opposite vectors.

A human-annotated benchmark dataset consists of *n* triplets containing pairs of words and their corresponding similarity scores $\langle w_i, w_j, h_{ij}\rangle$, where w_i , w_j are dictionary words, and h_{ij} is the score.

Let $h = (h_{i1}, h_{i2}, \dots, h_{iN})$ be a vector of human annotated benchmark datasets, and $m = (m_{i1}, m_{i2}, \dots, m_{iN})$, correspondingly, a vector of similarity scores calculated from word embeddings.

Then, the evaluation score for the corresponding embedding model, based on the selected benchmark, is calculated as *Spearman's* correlation ρ (see Eq. 2) between *h* and *m*.

Spearman p value can be any value satisfying$-1 \leq p \leq 1$, and the interpretation is that p values close to +1 indicate stronger relationship, while those closer to -1 indicate weaker relationship.

The *Spearman* correlation formula is:

$$p = 1 - \frac{6\sum d_i^2}{n(n^2-1)}, \tag{2}$$

where *n* – dataset length, *d* – difference between ranks of *h* and *m*.

The aggregated score of one-word embedding model p_{avg} is calculated (see Eq. 3) as:

$$p_{avg} = \frac{1}{n} * \sum_{i=1}^{n} p_i \tag{3}$$

where p_i is *Spearman* correlation value of specific benchmark, *n* – number of benchmarks.

In order to compare different embedding model types (*Word2vec*, *fastText*, *GloVe*). We can calculate an average score of all models' embeddings (see Eq. 4).

$$P_t = \frac{1}{n} * \sum_{i=1}^{n} p_{avg} \tag{4}$$

where P_t – an average score of all *t* word embeddings, *n* – number of *t* word embedding models.

5. Experiments and results

Experiments were carried out in a series of tasks:

1. Firstly, 7 selected evaluation benchmark datasets were adapted for the Lithuanian language.
2. Secondly, 50 word embeddings with different hyperparameter sets were trained on the accumulated experimental corpus.
3. Thirdly, the obtained word embedding models were evaluated using the adapted intrinsic evaluation benchmarks.
4. Finally, the resulting data were examined in order to determine the effect of different hyperparameters on benchmark evaluation results.

5.1. Adaptation of benchmark datasets

The selected 7 (see Chapter 4.1) evaluation benchmark datasets were adapted from English to Lithuanian language. There were 5610 word pairs at the beginning. After the adaptation process, 5573 word pairs remained. A total of 37 word pairs were discarded. The following problems were observed during the adaptation process:

1. Multiple words – in some cases, one-to-one word translation is not possible, when a two-word expression in the Lithuanian language is a correspondence to a single word in the English language. For example, for the word pair "computer – software", the Lithuanian translation would be "*kompiuteris – programinė įranga*". As we use vector-to-vector comparison, such word pairs were discarded.
2. The meaning of certain words has been shaped by American culture, e.g. words like soccer, football, and FBI. These are words that are commonly used in the US. Such words were replaced with Lithuanian synonyms
3. A few older words have undergone semantic changes as their meanings evolved. For example, the word pair "Arafat – terror", had a greater similarity back in history than it does now. Such pairs were discarded.
4. In some cases, both English words have the same meaning in the Lithuanian language, for example, the following pairs: "smart – intelligent", "happy – cheerful", "fast – rapid". Such word pairs as a result contained two equal words, and their scores were set to 10 (maximum similarity).

An excerpt of the adapted *SimLex999* dataset for the Lithuanian language is presented in Table 3.

Table 3
Excerpt of the adapted of *SimLex999* benchmark for the Lithuanian language

word1	word2	value	word1	word2	value
steak	meat	7.47	kepsnys	mėsa	8.2
nail	thumb	3.55	nagas	nykštys	4.5
band	orchestra	7.08	grupė	orkestras	7.6
book	bible	5.00	knyga	biblija	5.6

The first two columns contain an English word pair in its original form. The third column contains the human-generated similarity score. The fourth and fifth columns contain Lithuanian translations of English words and revalued Lithuanian word scores.

5.2. Word embedding model training

The following tools were used for word embedding training: python genism wrapper of *Word2vec*[1], *fastText* – official python library[2], *GloVe* - official library[3]. We used similar training parameters in order to be able to compare different word embeddings (see Table 4).

Table 4
Hyperparameters used for embedding model training

	Word2vec	*FastText*	*GloVe*
Architecture	CBOW, Skip-gram	CBOW, Skip-gram	Global word-word Co-occurrence matrix
Model training	Negative Sampling, hierarchical SoftMax	Negative Sampling	
Dimensions	100, 300, 500, 1000	100, 300, 500	100, 300
Window size	5	5, 10	5, 10
Minimum count	1, 2, 5	2, 5	2 ,5

A total of 50 (19 *Word2vec*, 20 *fastText*, 11 *GloVe*) word embeddings were created by applying different hyperparameter sets.

5.3. Word embedding model evaluation

All the trained embedding models were evaluated using the Spearman ρ correlation coefficient between human benchmark scores and vector space model scores. The results were grouped by different vector model types, characterized by different hyperparameter sets (see Table 4).

The best 4 and the worst 4 models ranked by the benchmark result average are presented correspondingly in Table 5 and Table 6. The first column in these tables indicates model name together with hyperparameter indication. The following labels are used: *N* – negative sampling, *S – SoftMax*, *CBOW – Continuous Bag of Words*, *SKIP – Skipgram*, *d* – dimension, *w* – window size, *m* – minimum count threshold, *i* – iteration count. The rest are benchmark names and Spearman ρ correlation scores. The last column shows aggregated *Spearman* p_{avg} correlation score of all the benchmarks.

[1] https://radimrehurek.com/gensim/models/word2vec.html
[2] https://github.com/facebookresearch/fastText/
[3] https://github.com/stanfordnlp/GloVe

Table 5
The best 4 word embeddings ranked by $\boldsymbol{p_{avg}}$ (Spearman aggregated score)

Model	*MEN*	*WS353*	*WS353R*	*WS353S*	*SimLex999*	*RG65*	*MTurk*	p_{avg}
FastText $_{SKIP}$ 300_d 5_w 5_m 5_i	0.718	0.693	0.539	0.779	0.412	0.733	0.684	0.651
FastText $_{SKIP}$ 300_d 5_w 2_m 5_i	0.717	0.679	0.513	0.771	0.41	0.737	0.682	0.644
FastText $_{SKIP}$ 100_d 5_w 5_m 5_i	0.712	0.681	0.544	0.785	0.388	0.721	0.678	0.644
Word2vec N_{SKIP} 300_d 5_w 1_m 5_i	0.711	0.679	0.507	0.749	0.422	0.766	0.66	0.642

Table 6
The worst 4 word embeddings ranked by $\boldsymbol{p_{avg}}$ (Spearman aggregated score)

Model	*MEN*	*WS353*	*WS353R*	*WS353S*	*SimLex999*	*RG65*	*MTurk*	p_{avg}
GloVe 300_d 10_w 1_m 5_i	0.657	0.584	0.426	0.689	0.378	0.706	0.614	0.579
GloVe 100_d 10_w 2_m 5_i	0.65	0.584	0.414	0.683	0.363	0.724	0.611	0.575
FastText $_{CBOW}$ 100_d 5_w 1_m 5_i	0.65	0.564	0.389	0.679	0.419	0.761	0.559	0.574
GloVe 100_d 10_w 1_m 5_i	0.647	0.588	0.432	0.673	0.356	0.709	0.609	0.573

Comparison between different types of embeddings (*Word2vec*, *fastText*, *GloVe*) was done by averaging p_{avg} by embedding type (see Eq. (4).

To be able to do score comparison, only embedding models with the same hyperparameters were used: dimensions (100, 300), window size (5), and minimum count (2, 5) (Figure 1).

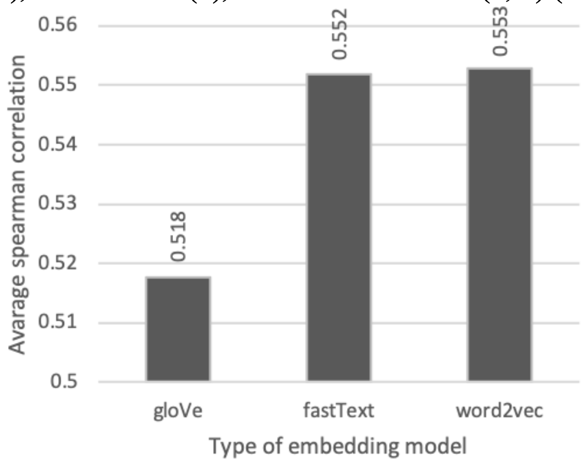

Figure 1: Aggregated *Spearman* ρ correlation scores over different model types P_t.

GloVe's word embedding model scores P_t were on average lower than those of *fastText* and *Word2vec*. The previous two were nearly identical, with a difference of only 0,001 between them.

Additionally, the experiment results were analyzed to determine whether a particular hyperparameter had a significant effect on the results. Following a thorough examination of all the hyperparameters, we discovered a correlation between the dimension value and the correlation results. (Figure 2).

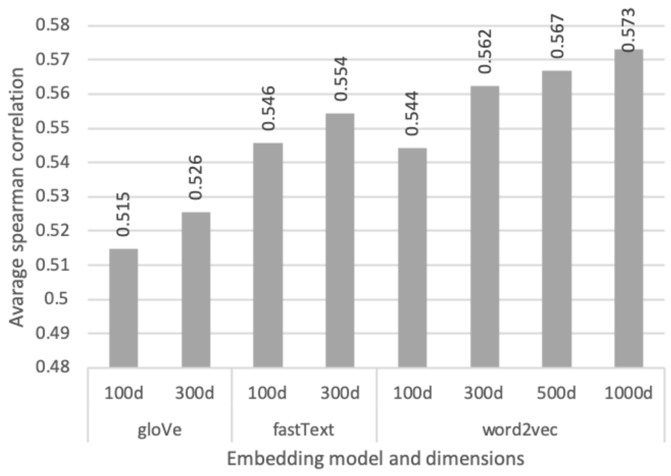

Figure 2: The correlation between vector size (*d*) hyperparameter and benchmark aggregated scores grouped by embedding model type.

Different dimension values for various embedding types had a significant effect on the results. The larger the dimension of the word embedding, the more accurate the results. As illustrated in Figure 3, as vector size increases, the model correlation score also increases.

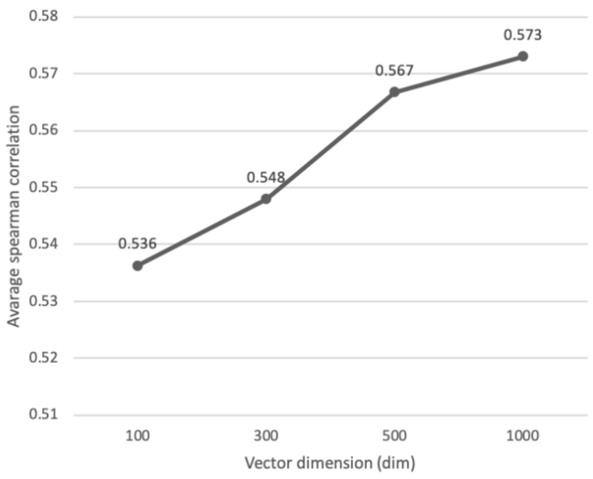

Figure 3: The correlation between vector size (dim) hyperparameter and benchmark aggregated scores.

We can use *Pearson* correlation score r (see Eq. 5) to see if there is correlation between values.

$$r = \frac{n(\sum x\,y) - (\sum x)(\sum y)}{\sqrt{[n\sum x^2 - (\sum x)^2]\,[n\sum y^2 - (\sum y)^2]}} \tag{5}$$

where n – number of models, x – dimension value. y – Spearman correlation value for a model.
$r = 0.918$, this indicates strong relationship between values.

6. Conclusions

This was the first attempt to adapt the most popular intrinsic similarity and relatedness benchmark datasets for the Lithuanian language. Despite reported challenges when adapting benchmarks to other languages, we proved, that this can be done even for morphology rich languages like Lithuanian.

The application of the adapted benchmark datasets for the evaluation of the embedding models, trained on an experimental corpus, showed, that *GloVe* model performed worse than *fastText* and *Word2vec*, judging by average benchmark results.

We also conclude, that for the intrinsic similarity and relatedness task, the dimension hyperparameter has a significant impact on the evaluation results, with larger word embedding dimension yielding better results.

In the future, we plan to adapt other types of embedding evaluation benchmarks, such as categorization and analogy testing, as well as extrinsic evaluation with POS tagging, named entity recognition (NER), and other NLP tasks. This would allow us to compare intrinsic and extrinsic evaluation methods. Also, we will continue to expand our corpus for future tests.

7. References

[1] Mikolov, Tomas, et al. "Efficient estimation of word representations in vector space." arXiv preprint arXiv:1301.3781 (2013).

[2] Pennington, J., Socher, R., Manning, C.D: Glove: global vectors for word representation. In: EMNLP, vol. 14, pp. 1532–1543 (2014)

[3] Hill, Felix, Roi Reichart, and Anna Korhonen. "Simlex-999: Evaluating semantic models with (genuine) similarity estimation." Computational Linguistics 41.4 (2015): 665-695.

[4] Bruni, Elia, Nam-Khanh Tran, and Marco Baroni. "Multimodal distributional semantics." Journal of artificial intelligence research 49 (2014): 1-47.

[5] Turian, Joseph, Lev Ratinov, and Yoshua Bengio. "Word representations: a simple and general method for semi-supervised learning." Proceedings of the 48th annual meeting of the association for computational linguistics. 2010.Mikolov, T., Chen,

[6] K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. In Proceedings of International Conference on Learning Representations (ICLR).

[7] Gladkova, A., Drozd, A., & Matsuoka, S. (2016). Analogy-based detection of morphological and semantic relations with word embeddings: what works and what doesn't. In Proceedings of the NAACL-HLT SRW (pp. 47–54). San Diego, California, June 12-17, 2016:

[8] Baroni, Marco, Georgiana Dinu, and Germán Kruszewski. "Don't count, predict! a systematic comparison of context-counting vs. context-predicting semantic vectors." Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (V

[9] Gimbutas, M. (1963), The Indo-Europeans: Archeological Problems. American Anthropologist, 65: 815-836. https://doi.org/10.1525/aa.1963.65.4.02a00030

[10] "Dictionary of the Lithuanian Language". 2002. Archived from the original on 2017-08-11. Retrieved April 19, 2018

[11] The biggest and busiest languages on Wikipedia, 2021. URL: https://www.pingdom.com/blog/the-biggest-and-busiest-languages-on-wikipedia/.

[12] Faruqui, Manaal, et al. "Problems with evaluation of word embeddings using word similarity tasks." arXiv preprint arXiv:1605.02276 (2016).

[13] Leviant, Ira, and Roi Reichart. "Separated by an un-common language: Towards judgment language informed vector space modeling." arXiv preprint arXiv:1508.00106 (2015).

[14] Kittask, Claudia, and Eduard Barbu. "Is Similarity Visually Grounded? Computational Model of Similarity for the Estonian language." Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2019). 2019.

[15] Kapočiūtė-Dzikienė, Jurgita, and Robertas Damaševičius. "Intrinsic evaluation of Lithuanian word embeddings using WordNet." Computer Science On-line Conference. Springer, Cham, 2018.

[16] Kapočiūtė-Dzikienė, Jurgita, Robertas Damaševičius, and Marcin Woźniak. "Sentiment analysis of lithuanian texts using traditional and deep learning approaches." Computers 8.1 (2019): 4.
[17] Stankevičius, Lukas, and Mantas Lukoševičius. "Testing pre-trained Transformer models for Lithuanian news clustering." arXiv preprint arXiv:2004.03461 (2020).
[18] Lai, Siwei, et al. "How to generate a good word embedding." IEEE Intelligent Systems 31.6 (2016): 5-14.
[19] Vytauto Didžiojo universitetas. Kompiuterinės lingvistikos centras. Dabartinės lietuvių kalbos tekstynas. 1998-2016, http://tekstynas.vdu.lt/tekstynas.
[20] Kutuzov, Andrey, and Elizaveta Kuzmenko. "To lemmatize or not to lemmatize: how word normalisation affects ELMo performance in word sense disambiguation." arXiv preprint arXiv:1909.03135 (2019).
[21] VDU vykdo ES finansuojamą projektą „SEMANTIKA 2“, 2020. URL: https://www.vdu.lt/lt/vdu-vykdo-es-finansuojama-projekta-semantika-2/
[22] Jivani, Anjali Ganesh. "A comparative study of stemming algorithms." Int. J. Comp. Tech. Appl 2.6 (2011): 1930-1938.
[23] Bruni, Elia, Nam-Khanh Tran, and Marco Baroni. "Multimodal distributional semantics." Journal of artificial intelligence research 49 (2014): 1-47.
[24] Finkelstein, Lev, Evgeniy Gabrilovich, Yossi Matias, Ehud Rivlin, Zach Solan, Gadi Wolfman, and Eytan Ruppin. (2002) Placing Search in Context: The Concept Revisited. ACM Transactions on Information Systems, 20(1):116-131.
[25] Agirre, Eneko, et al. "A study on similarity and relatedness using distributional and wordnet-based approaches." (2009).
[26] Agirre, Eneko, et al. "A study on similarity and relatedness using distributional and wordnet-based approaches." (2009).
[27] Hill, Felix, Kyunghyun Cho, and Anna Korhonen. "Learning distributed representations of sentences from unlabelled data." arXiv preprint arXiv:1602.03483 (2016).
[28] Radinsky, Kira, et al. "A word at a time: computing word relatedness using temporal semantic analysis." Proceedings of the 20th international conference on World wide web. 2011.
[29] Rubenstein, Herbert, and John B. Goodenough. "Contextual correlates of synonymy." Communications of the ACM 8.10 (1965): 627-633.
[30] Fast, dynamic translation tailored to your content needs, 2021. URL: https://cloud.google.com/translate