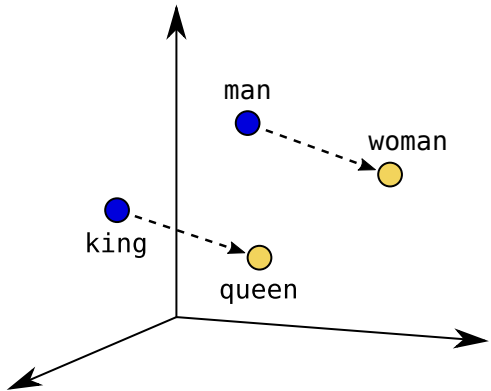
MUNI FI

Word Embeddings

Towards Fast, Interpretable, and Accurate Information Retrieval Systems

Vítek Novotný
witiko@mail.muni.cz
Math Information Retrieval Research Group
`https://mir.fi.muni.cz/`

October 15, 2020



Introduction

Why not use Transformers?

- Applications of Transformers [1] for full-text search are limited:
  - Documents are retrieved by keyword matching and top ten are reranked for speed. [2]
  - Due to $\mathcal{O}(w^2)$ space complexity of Attention, only short documents (QA) can be reranked.
- sBERT [3] can project short documents to semantic vector space, but it is supervised.
- Big Bird [4] makes Attention $\mathcal{O}(w)$ and could be used to rerank long documents.
- Until speed and space are improved, keyword matching determines search results.

How to improve keyword matching?

- Keyword matching similarity functions (TF-IDF, BM25) disregard word similarity.
- Shallow embeddings (word2vec [5, 6], GloVe [7], fastText [8]) measure word similarity.
- TF-IDF and BM25 can support soft keyword matching [9–12] using embeddings.

Introduction

Word Embeddings?

The quick brown **???** jumps over the lazy dog.

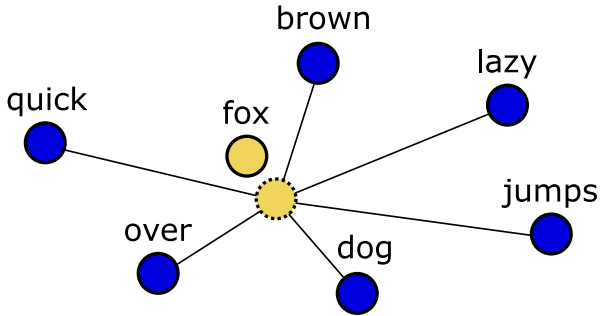


Figure: The training objective of word2vec and fastText CBOW models.

Introduction

Keyword Matching? I

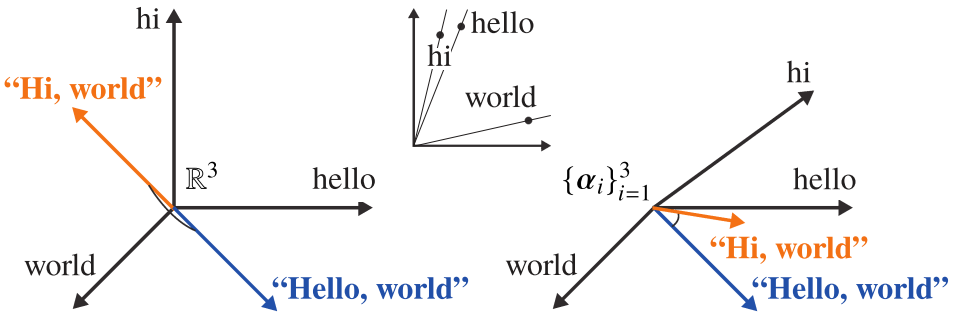
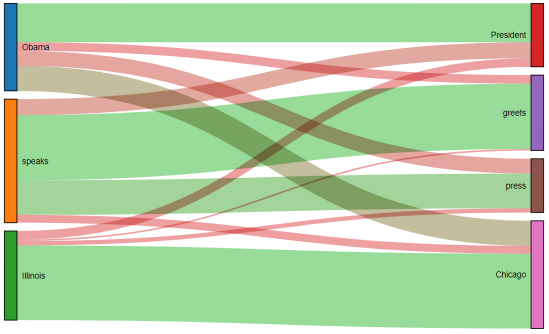


Figure: Keyword matching (left), word embeddings (middle), and soft keyword matching (right).

Introduction

Keyword Matching? II



Demo of soft keyword matching

Introduction

Keyword Matching? III

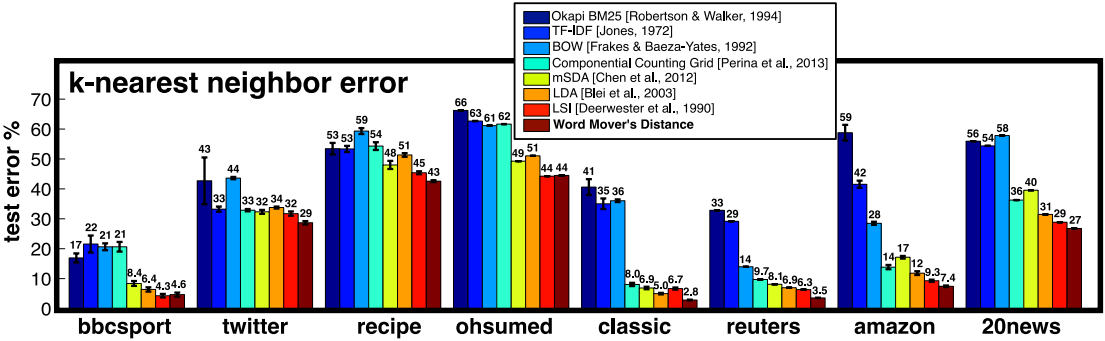


Figure: Text classification performance of soft keyword matching compared to other methods.

Introduction

How to improve keyword matching?

- TF-IDF and BM25 can support soft keyword matching [9–12] using embeddings.
- Our research group develops and maintains Gensim [13]:
  - Essential Python NLP library: 2.6k article citations and 11.2k stars on GitHub
  - Contains hardware-accelerated implementation of shallow word embeddings.
  - Perfect tool to prototype, implement, and evaluate enhanced word embeddings.

Position-dependence

Introduction

- Word2vec [5, 6] and fastText [8] CBOW models are trained to minimize the distance between the mean of context word embeddings and the masked word embedding.
- However, the position of words in context is not taken into account:
  1. "Unlike dogs, cats are **???**."
  2. "Unlike cats, dogs are **???**."
- Mikolov et al. [14] achieved +5% accuracy on English word analogy using position-dependent weighting [14, Section 2.2]. No implementation exists.

$$v_C = \frac{1}{|P|} \sum_{p \in P} d_p \qquad \Longrightarrow \qquad v_C = \frac{1}{|P|} \sum_{p \in P} d_p \odot u_{t+p}$$

- This summer, we drafted an implementation for Gensim. Open research questions:
  1. What is the optimal initialization?
  2. Are Mikolov's results reproducible?
  3. Is the speed practically useful?
  4. How does it improve end tasks?

Position-dependence

Word Analogy? (Source)

man:woman :: king:?

+	king	[0.30 0.70]
-	man	[0.20 0.20]
+	woman	[0.60 0.30]
	queen	[0.70 0.80]

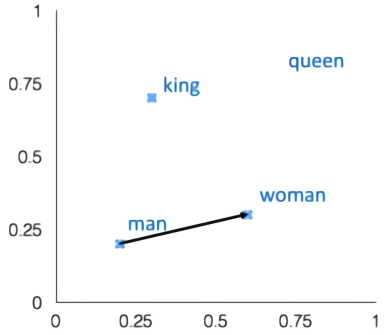


Figure: An example of the English word analogy task.

Position-dependence

What is the optimal initialization?

- FastText initializes d_p to $\text{Uniform}(\pm\frac{1}{\text{fanout}})$.
- The hidden layer receives $\frac{1}{|P|}\sum_{p\in P} d_p$, which tends to $\text{N}(0, \frac{\sigma^2}{|P|})$ by the CLT.
- To keep the same initial learning rate, we require $\frac{1}{|P|}\sum_{p\in P} d_p \odot u_{t+p} \sim \text{N}(0, \frac{\sigma^2}{|P|})$.
- We have several options for initialization:
  1. $d_p \sim \text{Uniform}(\pm\frac{1}{\text{fanout}}), u_{t+p} \sim d_p$.
  2. $d_p \sim \text{Uniform}(\pm\frac{1}{\text{fanout}}), u_{t+p} \sim 1$.
  3. $d_p \odot u_{t+p} \sim \text{Uniform}(\pm\frac{1}{\text{fanout}})$.
  4. $d_p \odot u_{t+p} \sim \text{N}(0, \sigma^2)$.
- Which initialization options are optimal?
  1. $\text{Var}[d_p \odot u_{t+p}] < \sigma^2$, leading to smaller initial learning rate.
  2. $\text{Var}[d_p \odot u_{t+p}] = \sigma^2$, but the gradients to d_p explode. (the only option we tried)
  3. $\text{Var}[d_p \odot u_{t+p}] = \sigma^2$, but only the square of positive uniform distribution is known [15].
  4. $\text{Var}[d_p \odot u_{t+p}] = \sigma^2$, and the square of normal distribution is known [16].

Position-dependence

$d_p \sim \textbf{Uniform}(\pm\frac{1}{\textbf{fanout}}), u_{t+p} \sim \textbf{Uniform}(\pm\frac{1}{\textbf{fanout}})$

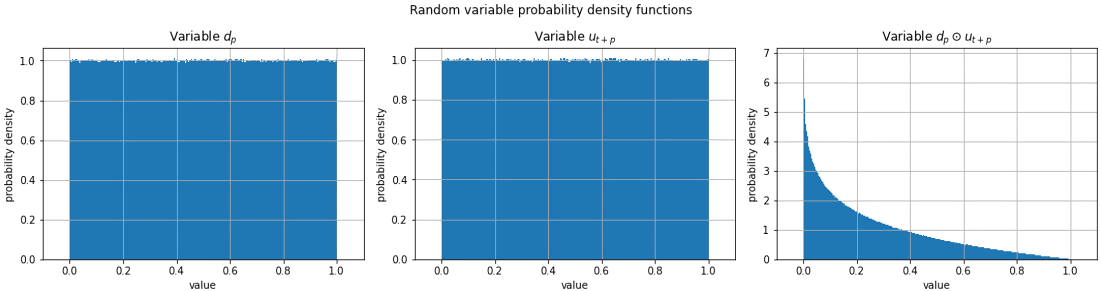


Figure: Probability densities of random variables for initialization option nr. 1.

Position-dependence

$d_p \odot u_{t+p} \sim$ **Uniform**$(\pm \frac{1}{\textbf{fanout}})$ **using the method of Ravshan [15]**

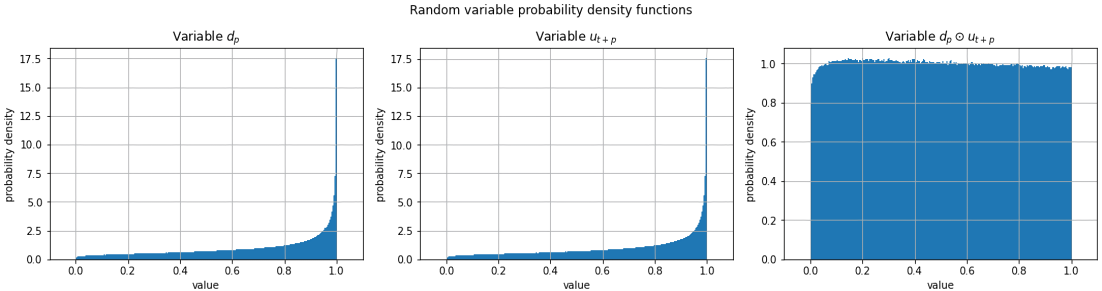


Figure: Probability densities of random variables for initialization option nr. 3.

Position-dependence

$d_p \odot u_{t+p} \sim \mathbf{N}(0, \sigma^2)$ using the method of Pinelis [16]

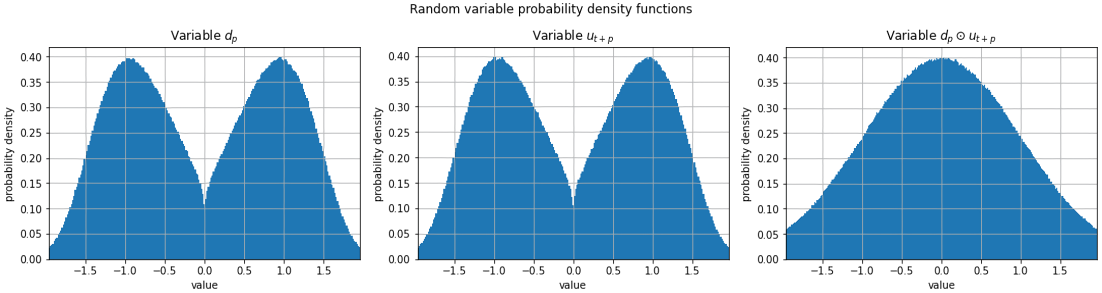


Figure: Probability densities of random variables for initialization option nr. 4.

Position-dependence

Are Mikolov's results [14] reproducible?

- We reproduced +10%* accuracy on English word analogy with these differences:
  - For speed, we used the 2017 English Wikipedia (2.4B words, ca 4% of Common Crawl).
  - For speed, we did only modeled unigrams, whereas Mikolov modeled up to 7-grams.

	Disabled	Enabled
Mikolov et al. [14]	80%	85%
Our results	62%	72%

Table: English word analogy task accuracies with and without position-dependent weighting

Position-dependence

Is the speed practically useful?

- Using full position-dependent weighting incurs 2–3× slowdown in training time.
- The speed is not practically useful: training a model for just 3 epochs without position-dependent weighting leads to +10%* accuracy on English word analogy.

Can we make the speed practically useful?

- No support in BLAS for $\odot$, abusing ?SBMV matrix op. incurs further slowdown.
- Replacing positional vectors with positional scalars removes all gains to accuracy.
- Reducing the dimensionality of positional vectors does the trick:

	Disabled	Enabled at 10d	Enabled at 300d
Accuracy*	62%	**72%**	**72%**
Training duration	2h 12m	**2h 29m**	5h 00m

Table: English word analogy task accuracies and training durations

Position-dependence

Reducing the dimensionality of positional vectors

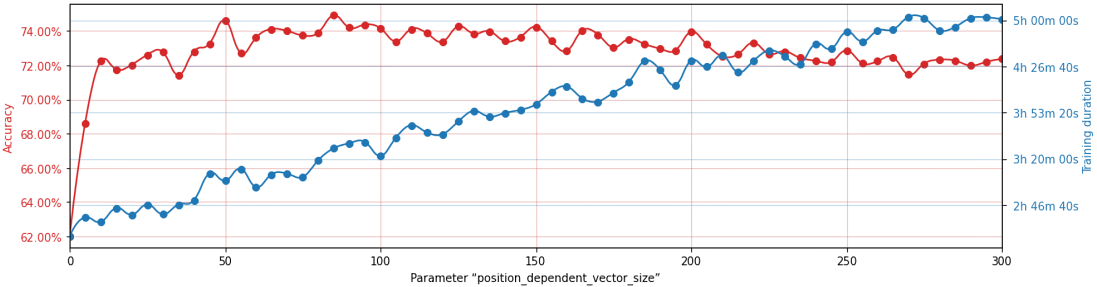


Figure: English word analogy task accuracies (red) and training durations (blue).

Position-dependence

What do the positional vectors look like?

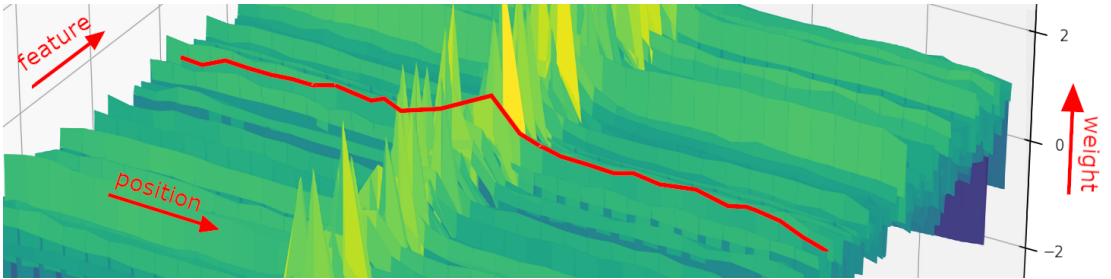


Figure: The positional vector features are normally distributed and continuous across positions.

Position-dependence

Can we use larger context windows now?

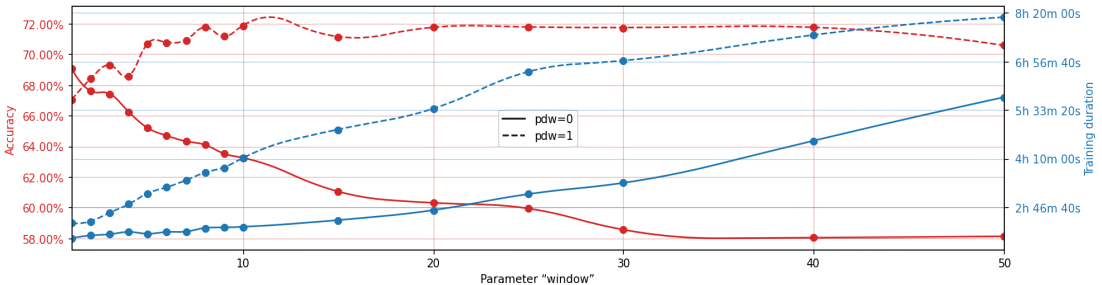
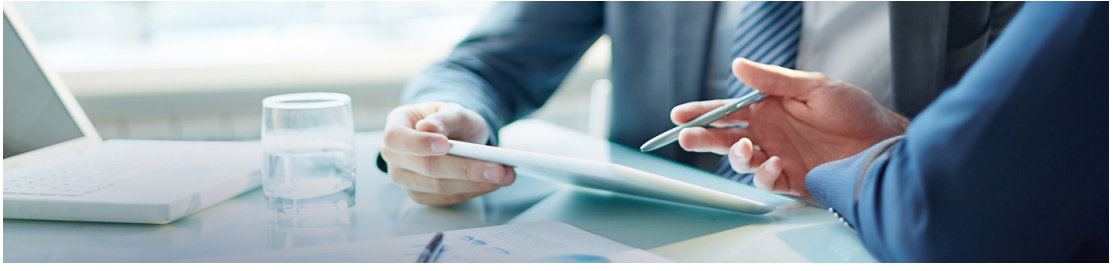


Figure: English word analogy task accuracies (red) and training durations (blue) for different context window sizes with position-dependent weighting enabled (dashed) and disabled (solid).

Position-dependence

How does it improve the end tasks?

- We have conducted experiments only on English word analogies.
- Possible end tasks:
  1. Text classification [17]
  2. Information retrieval [18, Section 4]
  3. Word similarities [8, Section 5.1]
  4. Language modeling [8, Section 5.6]
  5. Semantic text similarity [10]
  6. … other ideas?



Subword sizes

Parameter optimization

- Unlike word2vec [5, 6], fastText [8] embeds not only words, but also subwords.
- This speeds up training and allows inference of embeddings for unknown words.
- However, previous work reports optimal subword sizes only for English and German.
- Our experiments suggest:
  1. 5% improvement on Czech word analogy with optimal subword sizes over defaults.
  2. A fast method for estimating the optimal subword sizes from corpus statistics.
  3. Improvements on the language modeling end task.

Hyphenation

- Hyphenation splits words into subwords based on morphology or phonology.
- TEX's hyphenation algorithm [19] achieves perfect accuracy with tiny models. [20, 21]
- fastText embeds only subwords of fixed size and ignores morphology.
- Hyphenating fastText should decrease model size and speed up training.

Subword sizes

Subwords? (Source)

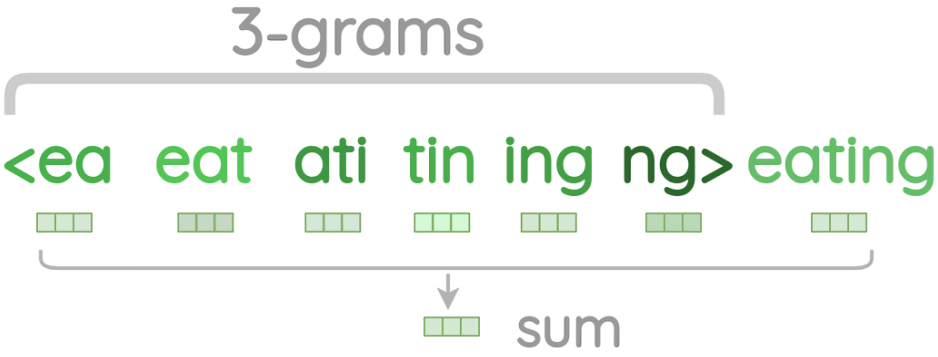


Figure: In word2vec, only the entire word is embedded (right). In fastText, n-grams are also embedded (left), which introduces weight sharing and enables inference for unknown words.

Subword sizes

Hyphenation? (Source)

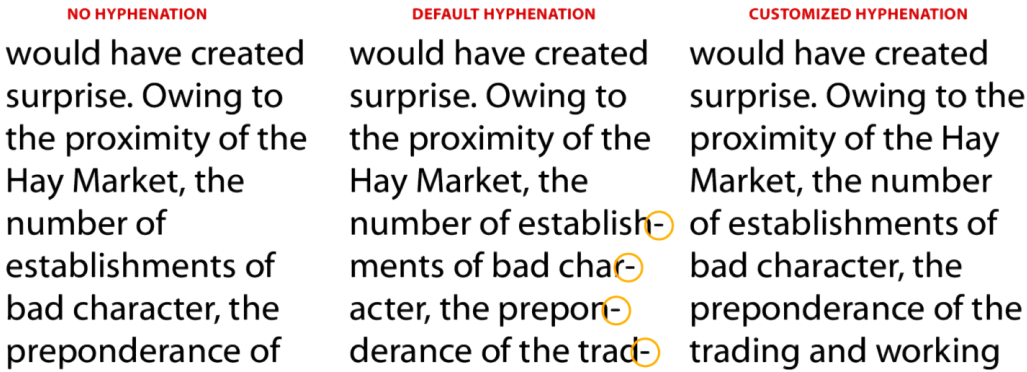


Figure: In typesetting, hyphenation enables justification without excessive spacing. To aid understanding/oration, hyphenation occurs at morpologically/phonetically meaningful points.

Conclusion

Sounds fun?

- Take a look at our bachelor's and master thesis topics:
  - Positional weighting of fastText word embeddings (bachelor's thesis, master's thesis)
  - Finding optimal *n*-gram sizes for fastText Model (bachelor's thesis, master's thesis)
  - … or come up with your own!

Bibliography I

[1] Ashish Vaswani et al. “Attention is all you need”. In: *Advances in neural information processing systems*. 2017, pp. 5998–6008.

[2] Wei Yang, Haotian Zhang, and Jimmy Lin. “Simple applications of BERT for ad hoc document retrieval”. In: *arXiv preprint arXiv:1903.10972* (2019).

[3] Nils Reimers and Iryna Gurevych. “Sentence-bert: Sentence embeddings using siamese bert-networks”. In: *arXiv preprint arXiv:1908.10084* (2019).

[4] Manzil Zaheer et al. “Big bird: Transformers for longer sequences”. In: *arXiv preprint arXiv:2007.14062* (2020).

[5] Tomas Mikolov et al. “Efficient estimation of word representations in vector space”. In: *arXiv preprint arXiv:1301.3781* (2013).

Bibliography II

[6] Tomas Mikolov et al. “Distributed representations of words and phrases and their compositionality”. In: *Advances in neural information processing systems*. 2013, pp. 3111–3119.

[7] Jeffrey Pennington, Richard Socher, and Christopher D Manning. “Glove: Global vectors for word representation”. In: *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. 2014, pp. 1532–1543.

[8] Piotr Bojanowski et al. “Enriching word vectors with subword information”. In: *Transactions of the Association for Computational Linguistics* 5 (2017), pp. 135–146.

[9] Grigori Sidorov et al. “Soft similarity and soft cosine measure: Similarity of features in vector space model”. In: *Computación y Sistemas* 18.3 (2014), pp. 491–504.

Bibliography III

[10] Delphine Charlet and Geraldine Damnati. “Simbow at semeval-2017 task 3: Soft-cosine semantic similarity between questions for community question answering”. In: *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*. 2017, pp. 315–319.

[11] Vít Novotný. “Implementation notes for the soft cosine measure”. In: *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. 2018, pp. 1639–1642.

[12] Vít Novotný et al. “Text classification with word embedding regularization and soft similarity measure”. In: *arXiv preprint arXiv:2003.05019* (2020).

[13] Radim Rehurek and Petr Sojka. “Software framework for topic modelling with large corpora”. In: *In Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*. Citeseer. 2010.

Bibliography IV

[14] Tomas Mikolov et al. “Advances in pre-training distributed word representations”. In: *arXiv preprint arXiv:1712.09405* (2017).

[15] S. K. Ravshan. *Factor Analysis and Uniform distributions*. 2018. URL: `https://ravshansk.com/articles/uniform-distribution.html` (visited on 10/15/2020).

[16] Iosif Pinelis. “The exp-normal distribution is infinitely divisible”. In: *arXiv preprint arXiv:1803.09838* (2018).

[17] Matt Kusner et al. “From word embeddings to document distances”. In: *International conference on machine learning*. 2015, pp. 957–966.

[18] Vít Novotný et al. “Three is Better than One. Ensembling Math Information Retrieval Systems”. In: *CEUR Workshop Proceedings*. ISBN: 1613-0073.

Bibliography V

[19] Franklin Mark Liang. *Word Hy-phen-a-tion by Com-put-er*. Tech. rep. Calif. Univ. Stanford. Comput. Sci. Dept., 1983.

[20] Petr Sojka and Ondřej Sojka. “The unreasonable effectiveness of pattern generation”. In: *Zpravodaj CSTUG* (2019).

[21] Petr Sojka and Ondrej Sojka. “Towards Universal Hyphenation Patterns.”. In: *RASLAN*. 2019, pp. 63–68.

MUNI
FACULTY
OF INFORMATICS