# **NATURAL LANGUAGE PROCESSING**

# **DIGITAL ASSIGNMENT**

**The Gensim Library:**

**Introduction**

Genism is a state of the art natural language tool mainly focussed on topic modelling, document indexing and similarity retrieval with large corpora.

GENSIM: GENERATE SIMILARITY

Gensim is the most robust, efficient and hassle-free piece of software to realize unsupervised semantic modelling from plain text.

The mind behind this amazing tool is **Radim Řehůřek**

**Who is Radim?**

Radim Řehůřek is a Czechoslovakian researcher and data scientist. He holds a master’s degree in computer science and a PhD with focus on High Performance computing and Natural Language Processing from Masaryk University Brno. He is currently the Director of R&D at RaRe Technologies, Bristol.

**GENSIM: Inception of the idea**

Radim Řehůřek was working on a project with The Czech Digital Mathematics Library. The aim of this project was to investigate, develop and apply techniques, methods and tools that would allow the creation of a suitable infrastructure and conditions for establishing the Czech Digital Mathematics Library (DML-CZ). The library consists of the relevant mathematical literature which has been published throughout history in the Czech lands.

Initially Radim Řehůřek wrote several python scripts to generate a short list of the **most similar articles to a given article.** Then he went ahead and tried to use the so called fancy **LATENT SEMANTIC METHODS.** He then realised that the libraries available to work with these methods were hard to understand and generally “NOT FUN” to work with (directly quoting Radim Řehůřek).

Radim Řehůřek then set out to change this and started working on a new library that would essentially reinvent the wheel by ensuring clarity, efficiency and Scalability.

This was the inception of Gensim

Radim Řehůřek then went on to release a paper along with his professor Petr Sojka titled “Software Framework for Topic Modelling with Large Corpora” which **introduced the world to Gensim.**

**Link:** https://radimrehurek.com/gensim/lrec2010\_final.pdf

This paper gives the initial idea that Radim Řehůřek had for Gensim and the crux of the paper is representative of Gensim works even today.

**The Challenges Faced:**

It is mentioned in the research paper that there were several NLP toolkits available before Gensim for similarity based tasks. They were NLTK, Apache UIMA and CleatTK. Radim Řehůřek analysed the shortcomings of these packages and had to overcome them.

1. These existing packages commonly offered Supervised learning functionality (Topic inference is an unsupervised task)
2. The existing packages were not scalable as they required the whole corpus to be in the memory while using sparse representations
3. The existing packages were created with specific domains in mind
4. The existing packages offered a GRAND UNIFIED FRAMEWORK covering a broad range of algorithms resulting in complex interfaces and dependencies

Rahim Řehůřek worked his whole PhD thesis around solving the scalability issue

**Link**: <https://radimrehurek.com/phd_rehurek.pdf>

**Quoting the paper:**

“” Scalability is achieved by building the semantic models in a constant amount of memory and distributing the computation over a cluster of autonomous computers, connected by a high-latency network. In addition, the novel LSA training algorithm operates in a single pass over the training data, allowing continuous online training over infinite-sized training streams””

Github link: <https://github.com/RaRe-Technologies/gensim>

Gensim is currently maintained by RaRe technologies and has a very active developer community continuously improving and adding new functionalities

The simple way to install gensim is:

pip install -U gensim

**THE MILLION DOLLAR QUESTION:**

**How come gensim is so fast and memory efficient? Isn’t it pure Python, and isn’t Python slow and greedy?**

Many scientific algorithms can be expressed in terms of large matrix operations (see the BLAS note above). Gensim taps into these low-level BLAS libraries, by means of its dependency on NumPy. So while gensim-the-top-level-code is pure Python, it actually executes highly optimized Fortran/C under the hood, including multithreading (if your BLAS is so configured).

Memory-wise, gensim makes heavy use of Python’s built-in generators and iterators for streamed data processing. Memory efficiency was one of gensim’s design goals, and is a central feature of gensim, rather than something bolted on as an afterthought.

**WHY GENSIM** ?

* All algorithms are memory-independent w.r.t. the corpus size (can process input larger than RAM, streamed, out-of-core),
* Intuitive interfaces
* Easy to plug in your own input corpus/datastream (trivial streaming API)
* Easy to extend with other Vector Space algorithms (trivial transformation API)
* Efficient multicore implementations of popular algorithms, such as online Latent Semantic Analysis (LSA/LSI/SVD), Latent Dirichlet Allocation (LDA), Random Projections (RP), Hierarchical Dirichlet Process (HDP) or word2vec deep learning.
* Distributed computing: can run Latent Semantic Analysis and Latent Dirichlet Allocation on a cluster of computers.
* Extensive documentation and Jupyter Notebook tutorials.

Gensim has a chat room <https://gitter.im/RaRe-Technologies/gensim> where you can ask your doubts, questions and have though provoking discussions with fellow developers.

**References:**

1. <https://radimrehurek.com/gensim/about.html>
2. <https://en.wikipedia.org/wiki/Gensim>
3. <https://rare-technologies.com/>
4. <https://cz.linkedin.com/in/radimrehurek>
5. <https://towardsdatascience.com/5-heroic-tools-for-natural-language-processing-7f3c1f8fc9f0>
6. <https://radimrehurek.com/gensim/lrec2010_final.pdf>
7. <https://github.com/RaRe-Technologies/gensim>