# LLMs4Subjects: LLM-based Automated Subject Tagging for a National Technical Library's Open-Access Catalog

**Andrea Delli**
s331998

**Vincenzo Avantaggiato**
s323112

**Michele Cazzola**
s323270

# Problem Statement

- TIBKAT technical documents
- GND Tags

# Subject tagging for technical documents

**Goal:** recommend the most relevant subjects from the GND collection for each technical record.

**TIBKAT**: collection of bilingual (*en*, *de*) technical documents from the Leibniz University's Technical Library

Types: Article, Book, Conference, Report, Thesis

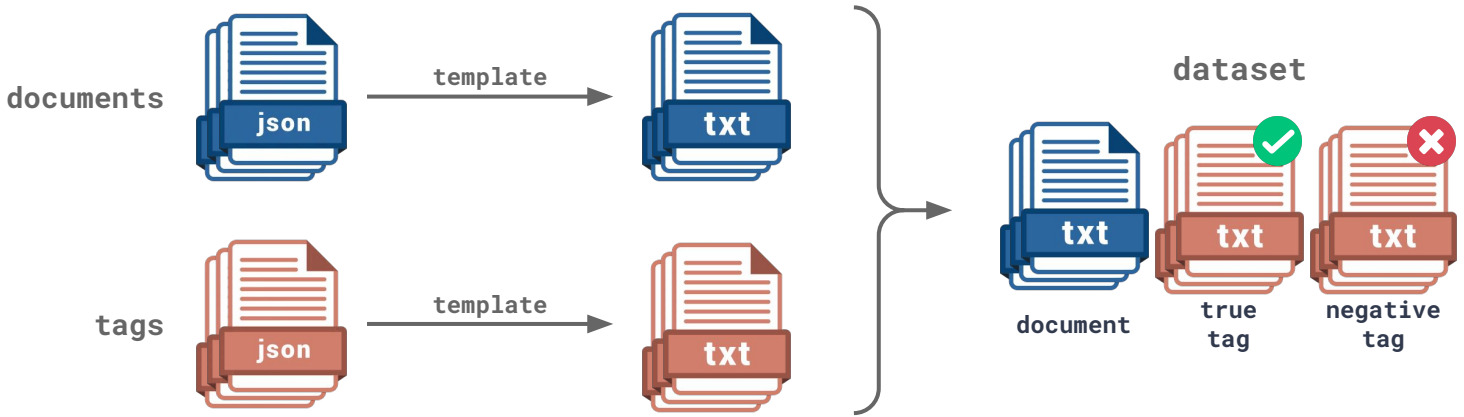
**GND (tags)**: international authority file used to catalog and link information about people, organizations, topics, and works.

Both documents and tags are pre-processed in JSON format for convenience, while documents are also splitted in **train**/**dev**/**test**.

# Dataset creation

We use a pre-defined **template** to generate a textual description from the JSON of documents and tags.

We create a dataset of **triplets** composed by **`(document text, true GND tag, negative GND tag)`**, used in experiments that require some training.



# Metrics

Results are evaluated by comparing the correct labels in the dataset and the ones assigned by us, using the metrics **precision@k**, **recall@k**, **F1@k** (*k=5, …, 50*).

Our prediction for each document: rank of 50 most similar GND tags

$$\text{Precision} = \frac{TP}{TP + FP}$$

$$\text{Recall} = \frac{TP}{TP + FN}$$

$$F1 = \frac{2 \times Precision \times Recall}{Precision + Recall}$$

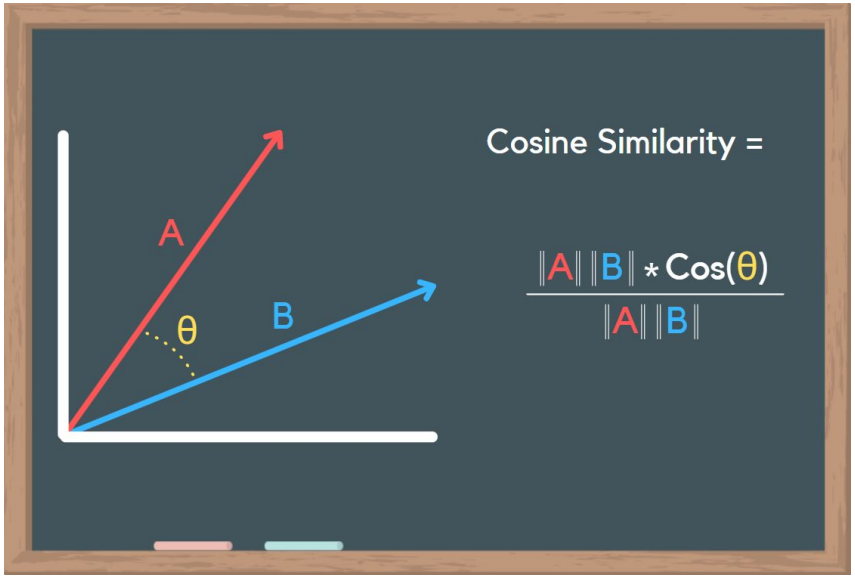
# Experiments

- Baseline models
- Fine-tuning embedding models
- MLP for embedding similarity

# 1st experiment: Compare using Cosine Similarity

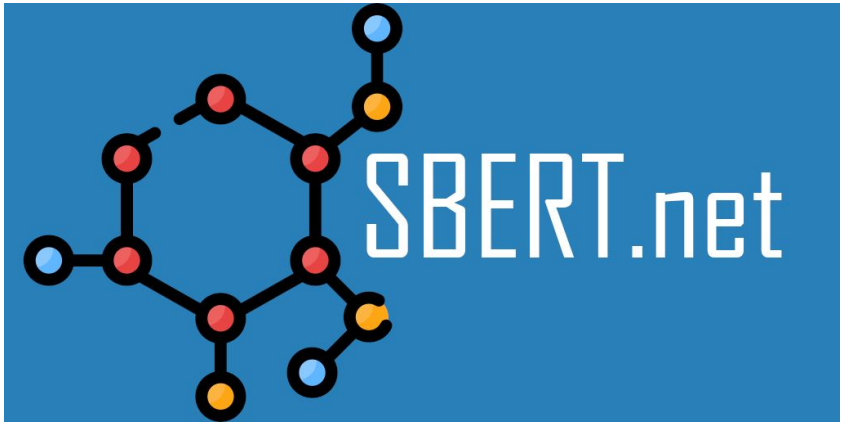
**Sentence embeddings** encode the semantic meaning and relationships between sentences, ensuring that semantically similar sentences have closely related representations.

To measure the distance between embeddings, we use **cosine similarity**.



# 1st experiment: Compare using Cosine Similarity

Since **BERT** is originally designed for token-level tasks, it does not naturally provide fixed-size sentence embeddings. To address this limitation, **Sentence-BERT** (SBERT) was developed, modifying BERT by introducing a siamese network architecture to derive semantically meaningful sentence embeddings.

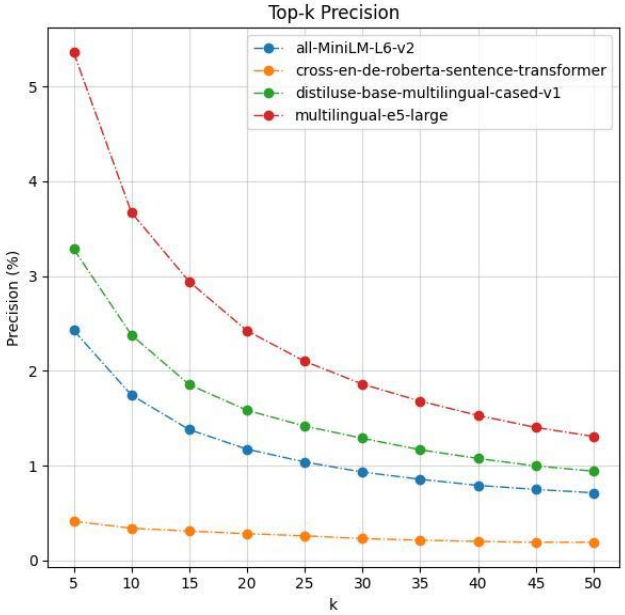


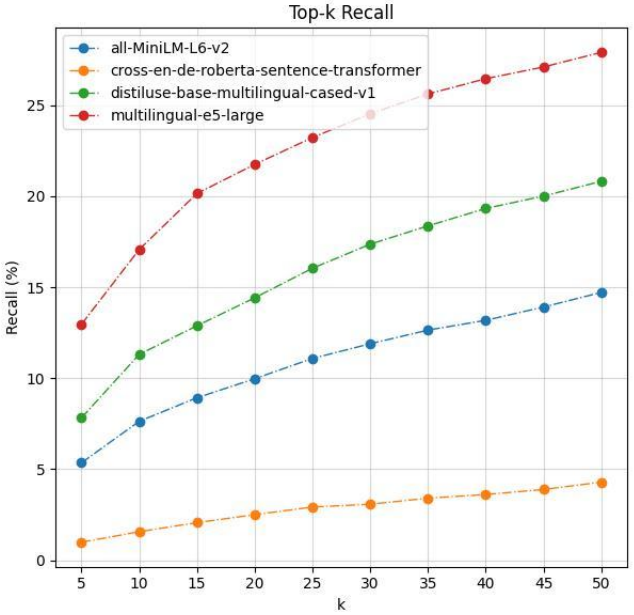
# 1st experiment: Compare using Cosine Similarity

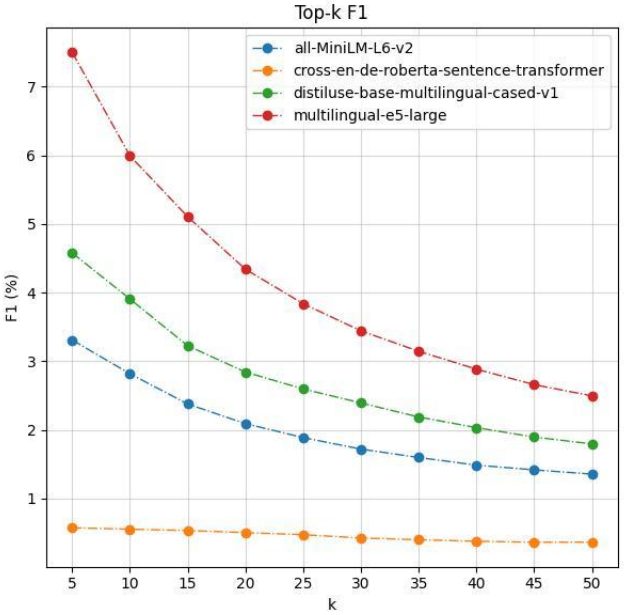
All the four embedding models we used belong to the **SBERT** family:

| Model | Embedding size | Parameters (M) | Latency (ms) |
|---|---|---|---|
| **all-MiniLM-L6-v2** | 384 | 22.7 | 8 |
| **distiluse-base-multilingual-cased-v1** | 512 | 135 | 8 |
| **cross-en-de-roberta-sentence-transformer** | 768 | 278 | 18 |
| **multilingual-e5-large** | 1024 | 560 | 55 |

# 1st experiment: Results



# 2nd experiment: Fine-tuning an embedding model

What if we fine-tune the models?

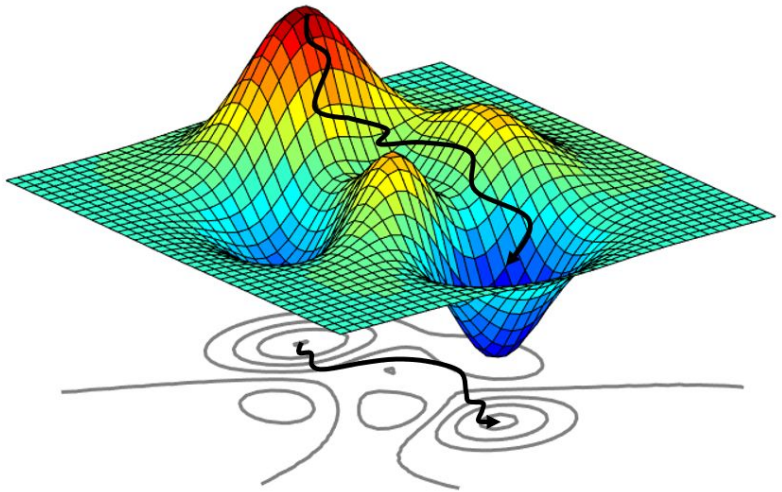
- fine-tuning is slow
- so, we only perform the experiment on **all-MiniLM-L6-v2**
- … for just 1 epoch

Which **loss** to use?

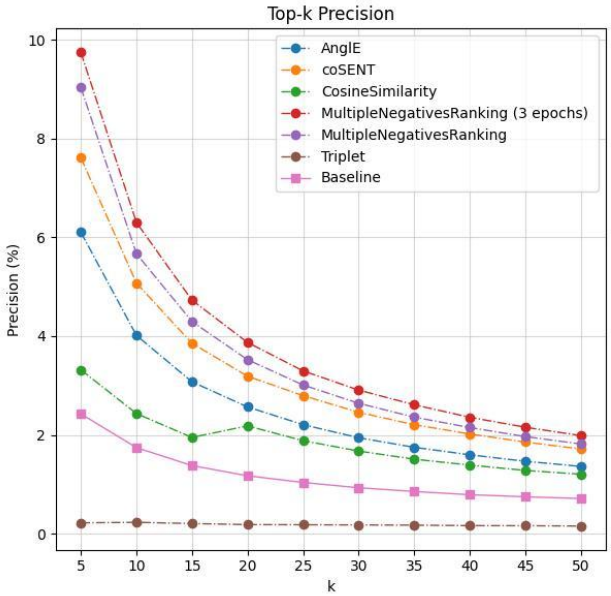
# 2nd experiment: Fine-tuning an embedding model

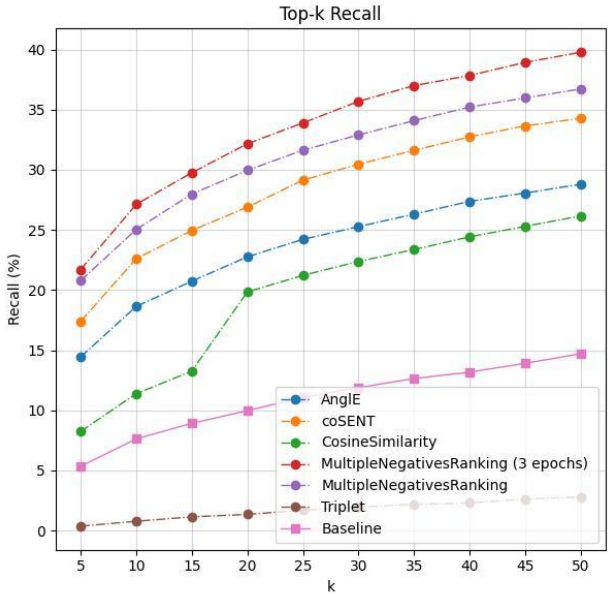
Losses:

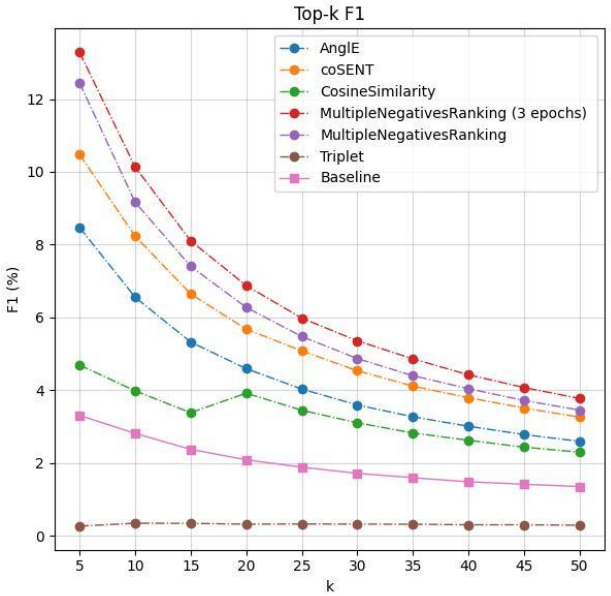
- TripletLoss
- CosineSimilarityLoss
- CoSENTLoss `loss = logsum(1+exp(s(k,l)-s(i,j))+exp...)`
- AnglELoss
- MultipleNegativesRankingLoss



# 2nd experiment: Results



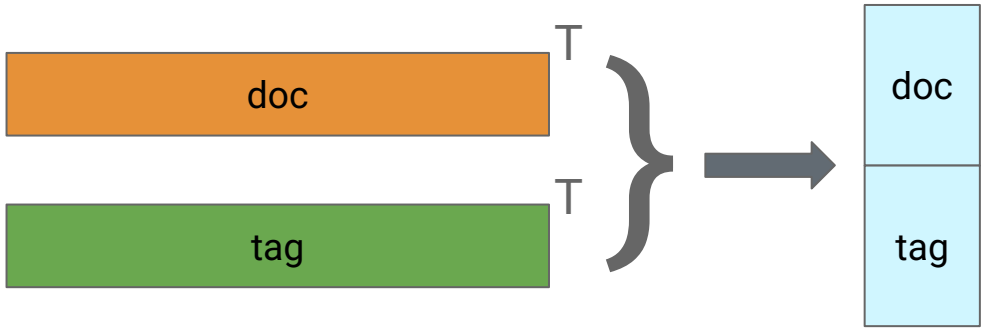



# 3rd experiment: MLP for embedding similarity

- Train a MLP on embeddings from *all-MiniLM-L6-v2*: binary task
- Goal: recognize how much a document is related to a given tag
- Output: tag each document and compute performances (inference)

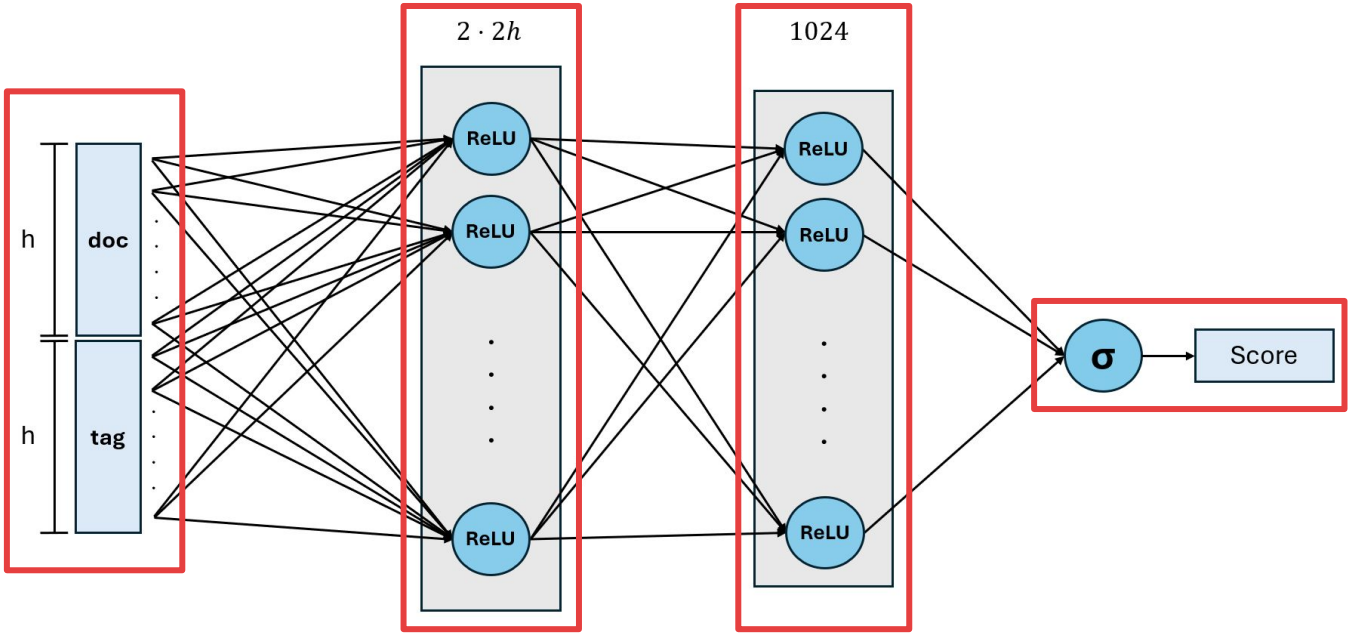
- Input: document embedding + tag embedding



# 3rd experiment: MLP for embedding similarity

**Architecture:**

- Input
- Hidden layers
- Output



# 3rd experiment: MLP for embedding similarity

**Loss function:** Binary Cross Entropy (BCE) loss

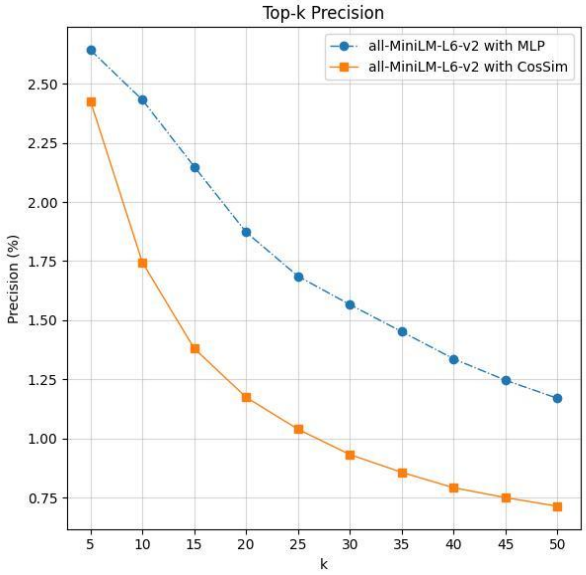
Positive GT

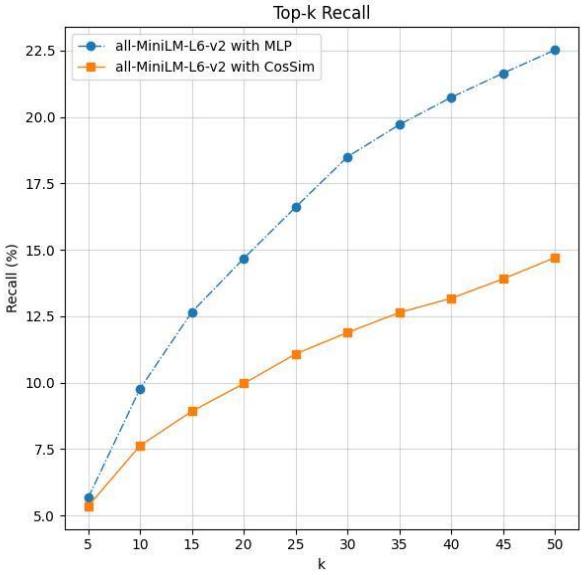
Negative GT

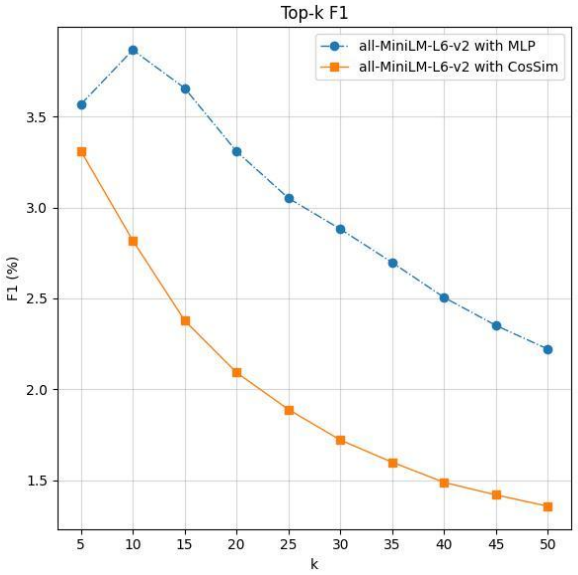
$$\mathcal{L} = -\sum_{i=1}^{N} \left[ y_i \log(\sigma(z_i)) + (1 - y_i) \log(1 - \sigma(z_i)) \right]$$

- $y_i$: ground truth for the i-th sample
- $z_i$: logit for the i-th sample
- N: number of samples

# 3rd experiment: Results



# Conclusions

- Best result: fine-tuning of *all-MiniLM-L6-v2* with *MultipleNegativesRanking* loss
- MLP: worse, but better than cosine similarity

**Future work**:

- fine-tune larger models (resource-intensive)
- train MLP on embeddings from fine-tuned models

# THANK YOU FOR YOUR ATTENTION!

**Andrea Delli**
s331998

**Vincenzo Avantaggiato**
s323112

**Michele Cazzola**
s323270