Magazine about news from the world of high-performance computing in Slovakia

2024

hpc focus

EURO
SLOVAKIA

ISBN 978-80-89871-25-4
ISSN 2729-9090

Editorial

Lucia
DEMOVIČOVÁ
Director of NSCC

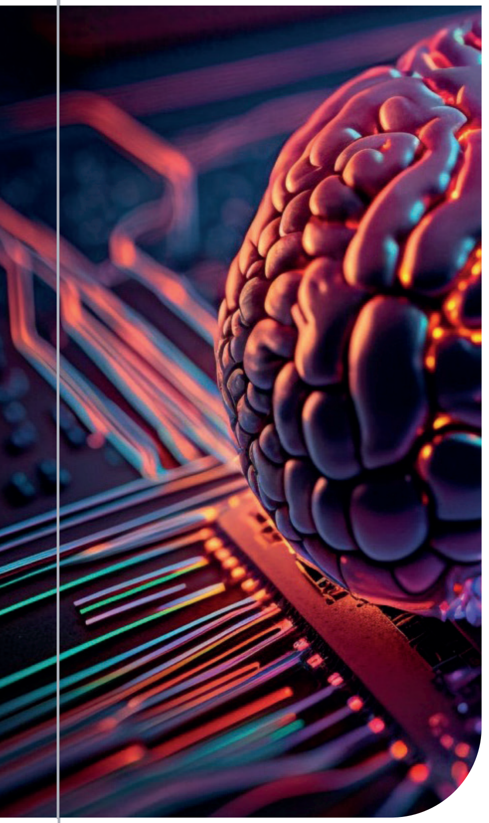
Dear readers and HPC enthusiasts,

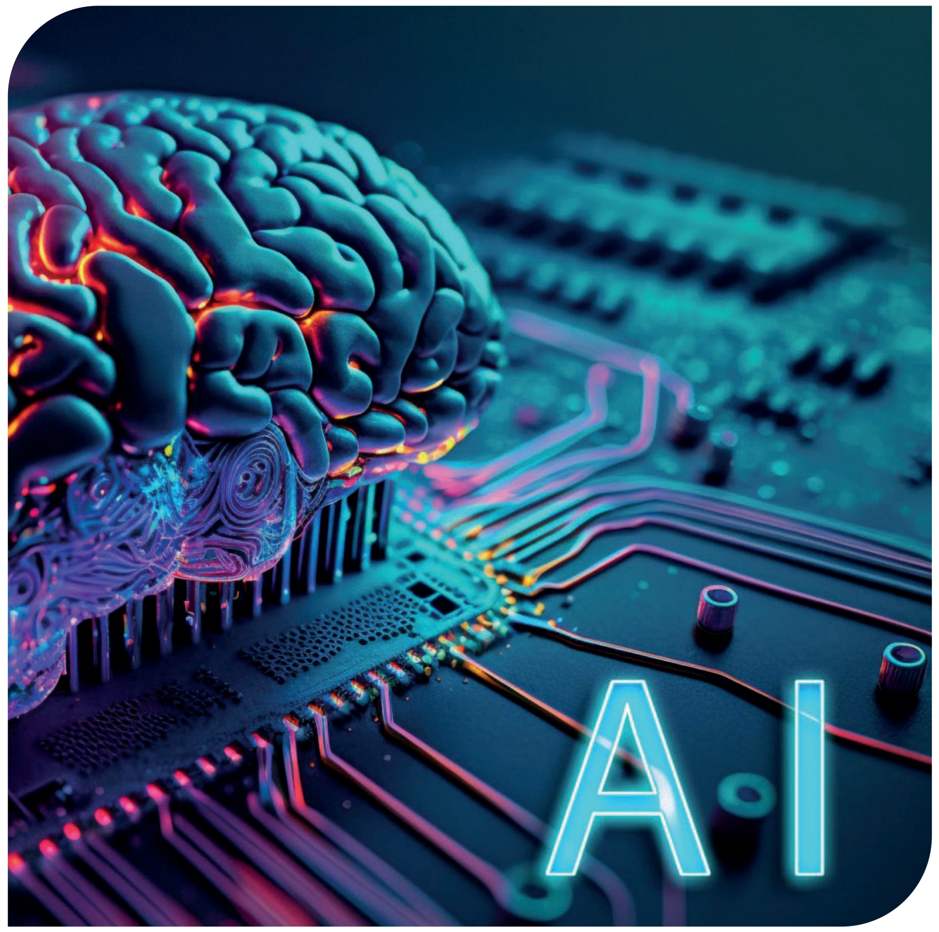
You are receiving the latest issue of HPC Focus magazine, which provides an overview of the most significant events and achievements in the field of HPC in Slovakia over the past year.

The Devana supercomputer is now fully operational, and we are delighted to see it serving a growing number of scientific disciplines and new users. In October 2023, we launched our first call for applications for standard access. As of this publication, we are already running our fourth such call. Altogether, we have distributed over 80 million CPU core-hours and nearly 300,000 GPU machine-hours to 48 projects. The demand for computing capacity, especially in accelerated environments, continues to grow, now surpassing the capabilities of our local infrastructure.

We are also continuously working to improve our services, keeping pace with current trends and making our users' work easier. Recently, we introduced a selection of tools for fine-tuning large language models on Devana. As AI remains a prominent topic, we are showcasing examples of successful projects in this issue that are related to this area. Additionally, we are expanding our portfolio of public courses to include topics related to AI and data processing.

Alongside our regular operations, we are also working on another major project—building the PERUN supercomputer. This system is expected to surpass Devana's capabilities by an order of magnitude and will position Slovakia more prominently on the HPC map of Europe. If the project is successfully implemented, PERUN will be handed over to the customer—CSČ SAV, v. v. i.—by the end





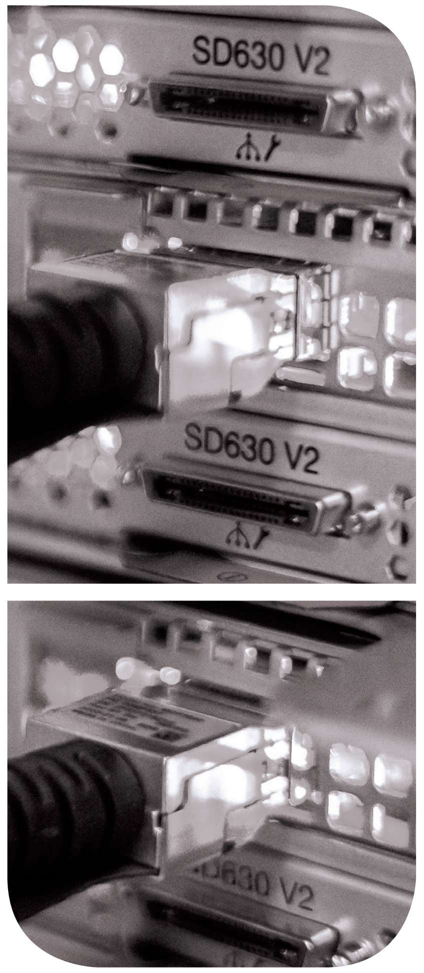
We are also continuously working to improve our services, keeping pace with current trends and making our users' work easier.

of 2025, with user access following shortly thereafter. This ambitious project is part of Slovakia's recovery and resilience plan and aims to place PERUN among the top ten most energy-efficient HPC systems in the Green500 ranking.

Slovakia is positioned to catch up with its neighbors and European partners in HPC development. However, achieving this goal requires more than just installing a supercomputer. In addition to providing computational power, it is crucial to engage, educate potential users, and tailor services and support tools to their needs. The foundation of this ecosystem has been laid through the efforts of the National Competence Centre, and the EuroCC and EuroCC 2 projects. We are confident that, with continued collaboration with our partners and NSCC members, we will keep advancing this development.

SUPERCOMPUTER DEVANA IS ALREADY IN FULL OPERATION.

CONTENT



Európska únia
Európsky fond regionálneho rozvoja
Operačný program VÝSKUM a VÝVOJ
Agentúra Ministerstva školstva, vedy, výskumu a športu SR pre štrukturálne fondy EÚ
Podporujeme výskumné aktivity na Slovensku
Projekt je spolufinancovaný zo zdrojov ES

01

NSCC NCC PRE HPC EUROHPC

Language Models: Modern Tools for Solving Complex Tasks

BIBIÁNA LAJČINOVÁ
MICHAL PITOŇÁK
PATRIK VALÁBEK

Artificial intelligence is currently one of the fastest-developing fields in information technology. Particularly noteworthy is the discipline of Natural Language Processing (NLP), which is an interdisciplinary field combining computer science, linguistics, mathematics, philosophy, and other areas. The tasks in this field include natural language understanding, natural language generation, sentiment analysis, information retrieval, and information extraction, among others. You can explore a particular use case and its solution in the following article Named Entity Recognition for Address Extraction in Speech-to-Text Transcriptions Using Synthetic Data [1] on the website of the National Competence Center for HPC. You will learn about a specific application of the Named Entity Recognition (NER) method to textual data containing addresses, with the goal of extracting the street name, house number, municipality name, and postal code using the optimization of the SlovakBERT language model. In another article, Leveraging LLMs for Efficient Religious Text Analysis, we focus on the creation of vector databases (embedding indices) from religious texts using both open-source and proprietary models, with the goal of efficiently retrieving document passages with relevant content (information retrieval).

The development of both applications utilized the computational resources of the HPC system Devana because optimizing language models requires substantial computational power. These models often contain billions of parameters, making it practically impossible to fine-tune them on standard personal computers. The Devana HPC system comprises 8 GPU compute nodes, each equipped with 4 nVidia A100 GPU cards, each with 40 GB of memory. Thanks to this setup, distributed training can be employed for very large models, either through data or model parallelism. Distributed training allows for the efficient distribution of computational load across multiple GPUs and/or compute nodes, thus accelerating the training of the models.

Language Models

The fundamental building blocks of NLP field are language models. These models "understand" and can even generate new text thanks to their architecture and extensive training on large amounts of textual data. They are built on deep neural networks, specifically utilizing a topology known as *transformers*. This technology was first introduced in 2017 in the paper *Attention is All You Need* [2], and it revolutionized the field of NLP. *Transformers* enable processing of sequential data, making them particularly well-suited for working with language and text (as well as other types of sequential data). Unlike traditional neural networks used for sequential data, such as recurrent networks, transformer-based models include an attention mechanism. This mechanism allows the network to determine which parts of the input to focus on

Artificial intelligence is currently one of the fastest-developing fields in information technology. Particularly noteworthy is the discipline of Natural Language Processing (NLP), which is an interdisciplinary field combining computer science, linguistics, mathematics, philosophy, and other areas.

and how individual parts of the input are related, thereby improving the prediction of the most likely output.

Another key component of natural language processing is tokenization. This process converts text into ”tokens”, which are numerical representations of words, parts of words, or individual characters, depending on the model. Tokens capture the context and meaning of the words and serve as input to the neural network of language models.

Other terms frequently encountered in the context of language models include:

- Embedding

  Vector representation of text (such as sentences or phrases) that captures meaning and context in numerical form.

- Fine-tuning

  The process of optimizing a pre-trained model for a specific task using a smaller dataset and (usually) adjusting only a subset of parameters. This process enhances the model's performance on particular tasks. The pre-trained model has already been trained on a vast amount of data, requiring significant computational resources. During fine-tuning, the model's parameters are further adjusted based on specific data (in smaller quantities) that were not included in the initial training.

Among the leading ”state-of-the-art” models is BERT (Bidirectional Encoder Representations from Transformers), developed by Google. This open-source model features only the encoder component and, despite its relatively simple architecture, is widely utilized for various tasks due to its flexibility. Another prominent model is GPT (Generative Pre-trained Transformer) from OpenAI. This multilingual model excels in generative tasks and can produce fluent, creative text. However, GPT is not publicly available and can only be accessed through an API (Application Programming Interface), as are other models from tech giants like Facebook and Microsoft. While these commercial models provide cutting-edge performance, they may not be suitable for everyone due to financial costs or when handling sensitive or proprietary data.

How to Train and Use a Language Model

For users of the HPC system Devana, our website offers a collection of tutorials and sample programs (in Python) for developing custom applications that leverage language models. These tutorials address both multilingual and Slovak language models, enabling users to either fine-tune them with their own data or use the pre-trained versions without modification. Each tutorial includes instructions for running the programs on the HPC system Devana using the Slurm management system, along with other pertinent details. Specific applications covered in the guides include:

- Sentiment Analysis Using BERT Model on Twitter Data

  See the following section.

- Prediction (Inference) Using a Pre-trained LLM Model for Question Answering

  This task involves using open-source models Mistral 7B Instruct [3] and Aya 101 [4] for question-answering. To test the accuracy of these models, we use open-source MedQA dataset [5], which includes questions and answers from the medical field. The goal is to demonstrate that large language models can be effectively used without fine-tuning.

- Fine-Tuning the Mistral 7B Instruct Model for Question Answering

  This task focuses on fine-tuning the Mistral 7B Instruct model [3] for question-answering. We use open-source MedQA dataset [5], which contains questions and answers from the medical field, for training the model. The goal is to enhance the model's ability to accurately answer medical-related questions, thereby improving its precision and relevance in this specific context.

- Fine-Tuning the DistilBERT Model for Named Entity Recognition (NER) with Hyperparameter Optimization Using Optuna

  In this task, we optimize the DistilBERT model [6] for named entity recognition (NER) in text. We use the publicly available CONLL2003 dataset [7], which includes annotations for various types of entities, such as persons (PER), locations (LOC), organizations (ORG), and others (MISC). We search for optimal training hyperparameters using the Optuna library, which allows systematic search for best values of learning rate, batch size, number of epochs, and more.

- **Creating a Vector Database Using Open-Source Embedding Models**
  In this task, we focus on creating a vector database using open-source embedding models. We use the BGE M3 model [8] to generate vector representations of texts (embeddings), enabling efficient text retrieval. The texts for generating embeddings come from the publicly available 20newsgroups dataset [9], which includes news articles and reports. These embeddings are used to build a vector database that facilitates rapid search and analysis of documents based on their content similarity.

We will analyze the first of the mentioned applications in more detail.

Sentiment Analysis Using the BERT Model on Twitter Data

Sentiment analysis is currently one of the common tasks for natural language processing algorithms. This task focuses on identifying and evaluating sentiment (i.e., "mood", typically classified as positive, negative, or neutral) in texts, which is useful for analyzing public opinion, customer feedback, moods, and trends across various social and business domains. Models like BERT have proven to be effective tools for this purpose. The data used for training these models is often sourced from social networks like Twitter and Facebook, as they reflect a wide range of user opinions and emotional expressions. Table 1 presents examples of tweets with different sentiments:

Tweet	Sentiment
Beautiful day! Sunny and everyone is smiling.	positive
I hate having to wait in long lines.	negative
Today is Tuesday.	negative

Since this task involves assigning a sentiment label to each observation, it is a classification problem with three classes. Therefore, we will use the BERT model with an output layer consisting of three neurons.

Next, we will demonstrate how to create a script for fine-tuning the BERT model using publicly available Twitter data.

Data Preparation, Model Selection, and Training

Before training the model, it is crucial to properly prepare the data. In our case, we use data in CSV format, consisting of tweets in English, each labelled with one of three possible sentiments: positive, neutral, or negative.

In this case, data preprocessing involves only removing rows with missing values.

```
df = pd.read_csv('Twitter_Data.csv')
df = df.dropna(subset=['category', 'clean_text'])  # Drop rows with NaN values
```

Since the data is in English, we have chosen the *BERT base model (uncased)* [10], a smaller version of BERT with fewer parameters than the *large* version. This model does not distinguish between uppercase and lowercase letters and includes its corresponding tokenizer.

The pre-trained model is loaded from the Hugging Face hub[1]. By using the *from pretrained* method, we load a model with pretrained weights, so only fine-tuning is needed, rather than training from scratch.

```
MODEL = "bert-base-uncased"
tokenizer = BertTokenizer.from_pretrained(MODEL, do_lower_case=True)
# Load pretrained model with 3 labels: positive, negative, neutral
model = BertForSequenceClassification.from_pretrained(MODEL, num_labels=3)
```

After splitting the data into training, validation, and test sets, all three sets are tokenized.

```
# Split data into train, validation, test sets
train_data, test_data = train_test_split(df.to_dict(orient='records'), test_size=0.2,
random_state=42)
train_data, val_data = train_test_split(train_data, test_size=0.1, random_state=42)

# Create DatasetDict for train, validation, test datasets
dataset_dict = DatasetDict({
```

[1] Hugging Face je platforma pre NLP a strojové učenie. Poskytuje široký výber pretrénovaných modelov a nástrojov, ktoré sú k dispozícii na ich webovej stránke https://huggingface.co.

```
    'train': Dataset.from_dict({'text': [item['clean_text'] for
item in train_data], 'label': [item['category_1'] for item in
train_data]}),
    'validation': Dataset.from_dict({'text': [item['clean_text']
for item in val_data], 'label': [item['category_1'] for item in
val_data]}),
    'test': Dataset.from_dict({'text': [item['clean_text'] for
item in test_data], 'label': [item['category_1'] for item in test_
data]})
})

tokenized_dataset = dataset_dict.map(tokenize_function,
batched=True)
```

Next, we define the training arguments, including the number of epochs, learning rate, and other parameters, and then proceed with fine-tuning the model using the *Trainer* method.

```
training_args = TrainingArguments(
    per_device_train_batch_size=64,
    per_device_eval_batch_size=64,
    output_dir="bert_twitter",
    num_train_epochs=5,
    weight_decay=0.1,
    evaluation_strategy="epoch",
    eval_steps=300,
    save_strategy="epoch",
    save_steps=300,
    load_best_model_at_end=True,
    eval_accumulation_steps=300,
    logging_steps=300
)

trainer = Trainer(
    model=model,
    args=training_args,
    train_dataset=tokenized_dataset["train"],
    eval_dataset=tokenized_dataset["validation"],
    tokenizer=tokenizer,
    data_collator=data_collator, # responsible for processing data
into expected format - batches
    compute_metrics=compute_metrics # function calculating metrics
to evaluate model's accuracy

trainer.train()
```

The fine-tuned model can be saved and used for predictions on new data.

```
model.save_pretrained("bert_sentiment_model")
```

Inference on new data can be performed using the *pipeline* method from the *transformers* library. The output includes not only the predicted class (sentiment) but also the probability, which indicates the model's confidence in assigning the text to the given class.

```
model = BertForSequenceClassification.from_pretrained("bert_sentiment_model")
sentiment_pipeline = pipeline("sentiment-analysis", model=model,
tokenizer=tokenizer, device=device) # inference initialization

text = "today is a beautiful day" # input text
result = sentiment_pipeline(text)
print(result)
```

Launching Training Using the Slurm Job Manager and Graphical Interface Through OpenOnDemand

Training the model is best carried out using the Slurm job manager, which provides access to the compute nodes of the HPC system Devana, rather than running it directly on one of the login nodes. Alternatively, you can use the graphical interface provided by the OpenOnDemand service, which is suitable for interactive work with code in Jupyter Notebooks. With this service, you can allocate the required computational resources for a specified period and then run model training and/or analyze results directly within the Jupyter Notebook environment.

The following shell script is used to initiate the fine-tuning of the BERT model for sentiment analysis on the HPC system using the Slurm job scheduler:

```
#!/bin/bash
#SBATCH --account=<your_project_number> # Project code
#SBATCH -o result.txt                   # Output file
#SBATCH -e error.txt                    # Error file
#SBATCH --gres=gpu:1                    # Request 1 GPU
#SBATCH --nodes=1                       # Request 1 node
#SBATCH --partition=gpu                 # Use GPU partition

module load singularity                 # Load singularity module
```

```
singularity exec --nv /storage-data/singularity_containers/pt-
2.3_llm.sif python3 sentiment_analysis_bert_train.py --batch_
size 64
```

The last line executes the Python script *sentiment_analysis_bert_train.py .py* using the Singularity container. The *–nv* parameter enables the use of GPU accelerators within the container. The container *pt2.3 _llm.sif* includes all necessary libraries, such as PyTorch, Transformers, Pandas, Numpy, and others.

Distributed Training

When training large language models with billions of parameters, parallelization is essential for efficient use of available computational resources and reduction of overall computation time. There are two main approaches to parallelization: data parallelism and model parallelism.

Data parallelism splits the training data into smaller chunks, which are processed simultaneously across multiple GPU accelerators. In contrast, model parallelism divides the model itself into smaller parts that are processed concurrently across multiple GPU accelerators or compute nodes. This approach is particularly useful and often necessary when the language model is too large to fit into the memory of a single GPU accelerator.

Models from the Hugging Face library support automatic model parallelism. In model parallelism, it is necessary to specify the devices on which the models will be loaded. By setting the device map argument to "auto", the model will be automatically distributed across the available resources.

```
model = BertForSequenceClassification.from_pretrained("bert_
sentiment_model", device_map = "auto")
```

Other options for distributed training are thoroughly described in the Hugging Face documentation. This page provides an overview of various parallelization techniques that can be used to efficiently distribute training across multiple GPU accelerators.

All tutorials and necessary auxiliary scripts for additional use cases, including instructions for execution, are available in our repository at github.com/NSCC-Slovakia. For further assistance or consultations regarding model training on the HPC system Devana, please feel free to schedule a personal or online meeting with us at NSCC.

REFERENCES

[1] Bibiána Lajčinová, Patrik Valábek, and Michal Spišiak. Named entity recognition for address extraction in speech-to-text transcriptions using synthetic data, 2024.

[2] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *CoRR*, abs/1706.03762, 2017.

[3] Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. Mistral 7b. *arXiv preprint arXiv:2310.06825*, 2023.

[4] Ahmet ¨Ust¨un, Viraat Aryabumi, Zheng-Xin Yong, Wei-Yin Ko, Daniel D'souza, Gbemileke Onilude, Neel Bhandari, Shivalika Singh, Hui-Lee Ooi, Amr Kayid, Freddie Vargus, Phil Blunsom, Shayne Longpre, Niklas Muennighoff, Marzieh Fadaee, Julia Kreutzer, and Sara Hooker. Aya model: An instruction finetuned open-access multilingual language model. *arXiv preprint arXiv:2402.07827*, 2024.

[5] Di Jin, Eileen Pan, Nassim Oufattole, Wei-Hung Weng, Hanyi Fang, and Peter Szolovits. What disease does this patient have? a large-scale open domain question answering dataset from medical exams. *Applied Sciences*, 11(14):6421, 2021.

[6] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *ArXiv*, abs/1*910.01108*, 2019.

[7] Erik F. Tjong Kim Sang and Fien De Meulder. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*, pages 142–147, 2003.

[8] Jianlv Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. Bge m3-embedding: Multi-lingual, multi-functionality, multi-granularity text embeddings through self-knowledge distillation, 2024.

[9] Ken Lang. Newsweeder: Learning to filter netnews. In *Armand Prieditis and Stuart Russell, editors, Machine Learning Proceedings 1995*, pages 331–339. Morgan Kaufmann, San Francisco(CA), 1995.

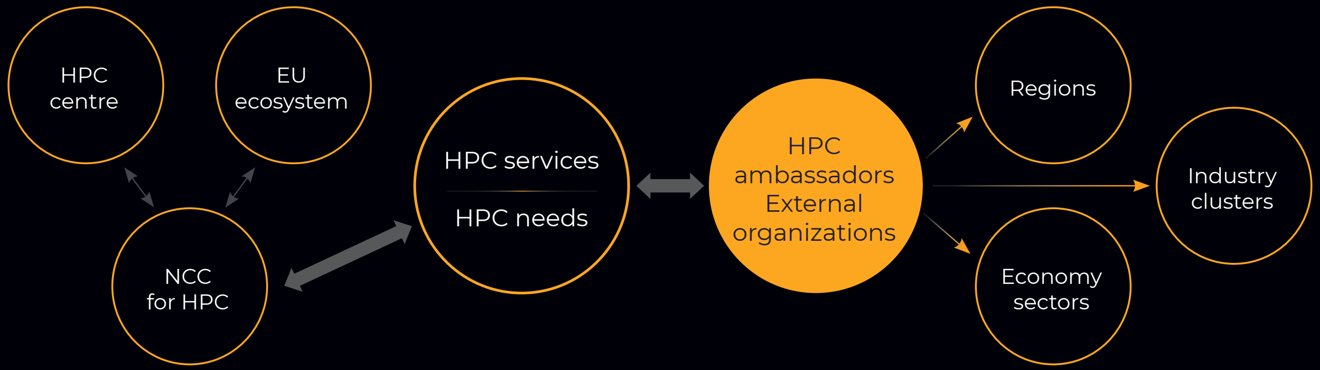
[10] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018.

When training large language models with billions of parameters, parallelization is essential for efficient use of available computational resources and reduction of overall computation time.

HPC Ambassador Program

WE COLLABORATE ACROSS SLOVAKIA

Since 2020, at the **National Competence Center for High-Performance Computing** (NCC), we have been reaching out to new enthusiasts and potential users of HPC technologies in Slovakia. In line with the goals of the European Union, we focus primarily on promoting the adoption of HPC in the private sector, among small and medium-sized enterprises, as well as larger industrial companies. We aim to ensure that Slovak companies can also implement advanced digital technologies and benefit not only from computing power but also from expertise, education, and opportunities to enhance their digital skills. NCC serves as a "one-stop shop" for high-performance computing in Slovakia. We have in-house experts who can be involved in pilot or proof-of-concept projects, organize lectures and courses for the public, and host workshops with both domestic and international experts.



Reaching out to a wide range of diverse companies, both large and small, with varying expectations, needs, and levels of preparedness is not easy. Therefore, as part of the development of the HPC ecosystem, we have decided to launch the HPC Ambassador Program. This program creates a network of contacts, bringing together organizations and key players in various Slovak regions and economic sectors.

The essence of this program is a mutually beneficial partnership between the competence center and individual organizations—ambassadors. Ambassadors are usually associations or clusters with a broad membership base and are well-established institutions in a specific region or area.

Through mutual communication and collaboration, we can better identify the needs of end clients/companies, their level of readiness to use HPC+ tools such as HPDA (high-performance data analytics), AI (artificial intelligence), or advanced numerical simulations. As a result, we can more effectively tailor our services and target communication, such as information on grant opportunities, educational possibilities within the European EuroCC network, or how to gain machine time in Slovakia and Europe. Here's how the collaboration looks:

INITIAL CONSULTATION NEEDS IDENTIFICATION

We analyze the structure and needs of your members, as well as the processes within your organization. This is a key step in tailoring our cooperation to your unique needs.

IDENTIFICATION OF POTENTIAL SME CANDIDATES

Using a simple survey matrix and a digital audit, we identify potential candidates among your members who could benefit from our HPC services.

TAILORED WORKSHOP

For specific candidates, we prepare a workshop tailored to their needs and goals, helping them better understand the potential for implementing specific HPC technologies.

FREE OPEN SCIENCE, POC PROJECT OR COMMERCIAL SOLUTION

We support open research and development by providing expert support and computing resources for pilot and proof-of-concept projects. If the project is successful, it can continue as a commercial solution with a positive impact on the company and its market competitiveness.

SYNERGIES | SHARING KNOW-HOW MEMBERSHIP IN ADVISORY BOARD

Ongoing sharing of best practices among the involved entities, creating mutual marketing potential. Additionally, we offer the opportunity to join the Advisory Board for collaboration with the private sector at the National Supercomputing Center and thus influence strategies and the development direction of HPC technologies in Slovakia.

We are very proud that shortly after launching the program, we already have the opportunity to collaborate with our ambassadors:

Innovation Center Inovia

Slovak Chamber of Commerce and Industry – Bratislava Chamber

Slovak Clusters Union

IT Valley Košice

We also collaborate with the Regional Centers of the Ministry of Investment, Regional Development, and Informatization of the Slovak Republic.

MINISTRY OF INVESTMENTS, REGIONAL DEVELOPMENT AND INFORMATIZATION OF THE SLOVAK REPUBLIC

```
    mirror_mod.use_z = False
elif _operation == "MIRROR_Z":
    mirror_mod.use_x = False
    mirror_mod.use_y = False
    mirror_mod.use_z = True

    #selection at the end -add back the deselected mirror modifier object
mirror_ob.select= 1
modifier_ob.select=1
bpy.context.scene.objects.active = modifier_ob
print("Selected" + str(modifier_ob)) # modifier ob is the active ob
    #mirror_ob.select = 0
```

The key selection criterion is the ability to demonstrate how the applicant plans to use the acquired experience and knowledge in HPC/HPC+.

Launch of the mobility programme HPC Fellowship

What is mobility programme HPC Fellowship?

The **HPC Fellowship** mobility programme which is coordinated by National Supercomputing Centre, is an initiative from the Digital Transformation Action Plan of Slovakia for 2023 – 2026. **It is designed for students in their second and third degree of studies, who are interested in gaining experience in the administration of HPC systems or in the development and deployment of highly parallel HPC applications in the so-called HPC+ (AI, HPDA, simulations).** The programme is open to students from various fields of study, including computer science, physics, engineering, biology, economics, but also social sciences, linguistics and many others. In cooperation with the participating host organizations, NSCC will publish the current framework topics for mobility projects.

The key selection criterion is the ability to demonstrate how the applicant plans to use the acquired experience and knowledge in HPC/HPC+ in further study or in their research area. The programme is financed by funds from the Ministry of Investments, Regional Development and Informatization of the Slovak Republic and its launch is planned for September 2024. **The mobility programme does not cover participation in semester courses at the host institution.**

GENERAL INFORMATION ABOUT THE PROGRAMME

- Duration: 1 to 3 months
- Host institutions: European Supercomputing Centres. During the first year of the programme will mainly focus on partner centres in the Czech Republic (IT4Innovations – VŠB Technical university of Ostrava), Austria (VSC – Vienna Scientific Cluster), Poland (Academic Computer Centre CYFRONET of the AGH University of Krakow) and Italy (CINECA, Bologna).
- Scholarship: Up to 4 000 € per month to cover travel costs, accommodation costs, food allowance, insurance costs, etc.

BENEFITS OF THE HPC FELLOWSHIP

1. Access to state-of-the-art facilities
   Students will have the opportunity to work in some of Europe's supercomputing centres, where they will gain hands-on experience with cutting-edge technologies and applications for HPC administration or applications in HPC+ areas.
2. Mentoring and networking
   Students will work with leading HPC experts and researchers and gain insight into the latest trends and challenges in the field. This experience is invaluable for building professional networks and exploring future career opportunities.

The programme is open to students from various fields of study, including computer science, physics, engineering, biology, economics, but also social sciences, linguistics and many others.

3. Academic growth

   Thanks to the scholarship and professional guidance, students will expand their horizons in the chosen field beyond the scope of regular studies, experience independent work on specific tasks and increase their competences.

Application process

The HPC Fellowship mobility programme is intended for master's degree and doctorate's degree students of Slovak universities who demonstrate an interest in HPC/HPC+. The application is available online on the website of the National Supercomputing Centre (nscc.sk) and is open throughout the academic year. Applicants must submit a curriculum vitae, transcript of records, and a motivation letter stating how they plan to use HPC in their final thesis or research areas.

Applications are evaluated by a selection committee consisting of representatives from the applicant's university and representatives from the National Supercomputing Centre. Priority is given to projects that demonstrate innovation, interdisciplinary collaboration and the potential for a significant contribution to the applicant's field of study.

02

HPC Applications

hpc focus

SEMI-SUPERVISED LEARNING IN AERIAL IMAGERY: IMPLEMENTING UNI-MATCH WITH FRAME FIELD LEARNING FOR BUILDING EXTRACTION

Patrik Sabol
Bibiána Lajčinová

Building extraction in GIS (geographic information system) is pivotal for urban planning, environmental studies, and infrastructure management, allowing for accurate mapping of structures, including the detection of illegal constructions for regulatory compliance. Integrating extracted building data with other geospatial layers enhances the understanding of urban dynamics and spatial relationships. Given the scale and complexity of these tasks, there is a growing need to automate building extraction using deep learning techniques, which offer improved accuracy and efficiency in handling large-scale geospatial data.

State-of-the-art image segmentation models primarily output in raster format, whereas GIS applications often require vector polygons. One such method to meet this requirement is Frame Field learning, which addresses the gap between raster format outputs of image segmentation models and the vector format needed in GIS. This approach significantly enhances the accuracy of building vectorization by aligning with ground truth contours and provide topologically clean vector objects.

These models are trained using a '*supervised learning*' method, necessitating a large amount of labeled examples for training. However, obtaining such a significant volume of data can be extremely challenging and expensive. A potential solution to this problem is '*semi-supervised learning,*' a method that reduces reliance on labeled data. In semi-supervised learning, the model is trained with a mix of a small set of labeled data and a larger set of unlabeled data. Hence, the goal of this collaboration between the Slovak National Competence Center for High-Performance Computing and Geodeticca Vision s.r.o. was to identify, implement, and evaluate an appropriate semi-supervised method for Frame Field learning.

Methods

FRAME FIELD LEARNING

The key idea of the frame field learning [1] is to help the polygonization method in solving ambiguous cases caused by discrete probability maps (output from image segmentation models). This is accomplished by introducing an additional output to the neural network of image segmentation, namely a frame field (see. Fig. 1), which represents the structural features and geometrical characteristics of the building.

Ing. Patrik Sabol, PhD. works as a Computer Vision Engineer at Geodeticca Vision.
He specializes in processing and analyzing geospatial data to create cutting-edge machine learning models.

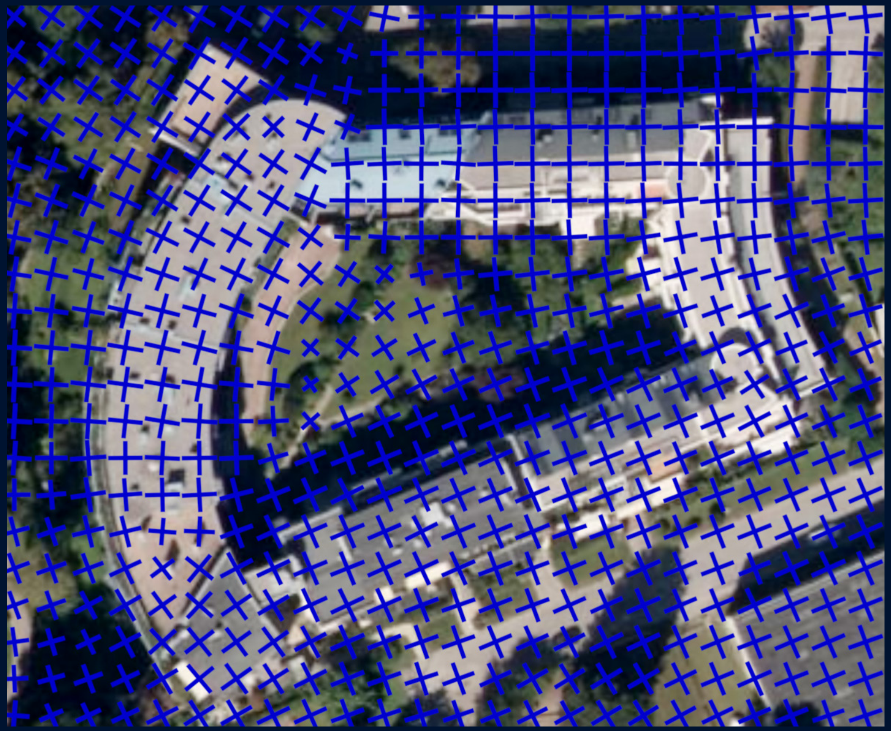


FIGURE 1

Visualization of the frame field output on the image from training set [1].

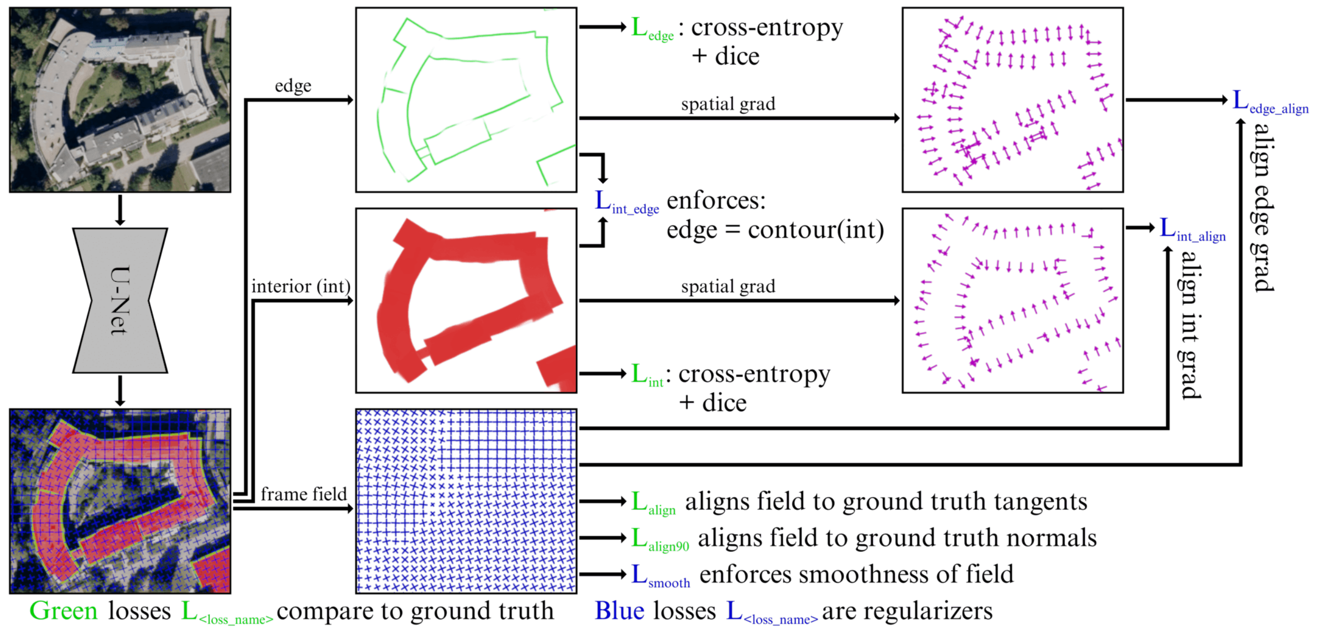
FRAME FIELDS

Frame field is a 4-PolyVector field that assigns four vectors to each point on a plane. Specifically, the first two vectors are constrained to be opposite to the other two, meaning each point is assigned a set of vectors {u, −u, v, −v}. This approach is particularly necessary for buildings, as they are regular structures with sharp corners, and capturing directionality at these sharp corners requires two directions.

FRAME FIELD LEARNING

The learning process of frame fields can be summarized as follows:

1. The network's input is a 3×H×W RGB image.
2. To generate a feature map, any deep segmentation model could be used, such as U-Net, which is then processed to output detailed segmentation maps.
3. The training is supervised with ground truth rasterized polygons for interiors and edges, utilizing a mix of cross-entropy and Dice loss for accurate segmentation.


U-Net
edge
interior (int)
frame field
L_{edge}: cross-entropy + dice
spatial grad
L_{int_edge} enforces: edge = contour(int)
spatial grad
L_{int}: cross-entropy + dice
L_{edge_align} align edge grad
L_{int_align} align int grad
L_{align} aligns field to ground truth tangents
$L_{align90}$ aligns field to ground truth normals
L_{smooth} enforces smoothness of field
Green losses $L_{<loss\_name>}$ compare to ground truth
Blue losses $L_{<loss\_name>}$ are regularizers

4. To train the frame field, three losses are used:
   - L_{align} – enforces alignment of the frame field to the tangent direction.
   - $L_{align90}$ – prevents the frame field from collapsing to a line field.
   - L_{smooth} – measures the smoothness of the frame field.
5. Additional losses, regularization losses, are introduced to maintain output consistency, aligning the spatial gradients of the predicted maps with the frame field.

FIGURE 2

Diagram of the frame field learning [1].

VECTORIZATION

The vectorization process transforms classified raster images into vector polygons using a polygonization method using the Active Skeleton Model (ASM). The principle of this algorithm is the iterative shifting of the vertices of the skeleton graph to their ideal positions. This method optimizes a skeleton graph – a network of pixels outlining the building's structure – created by a thinning method applied on a building wall probability map. The iterative shifting is controlled by a gradient optimization method aimed at minimizing an energy function, which includes specific components related to the structure and geometry being analyzed:

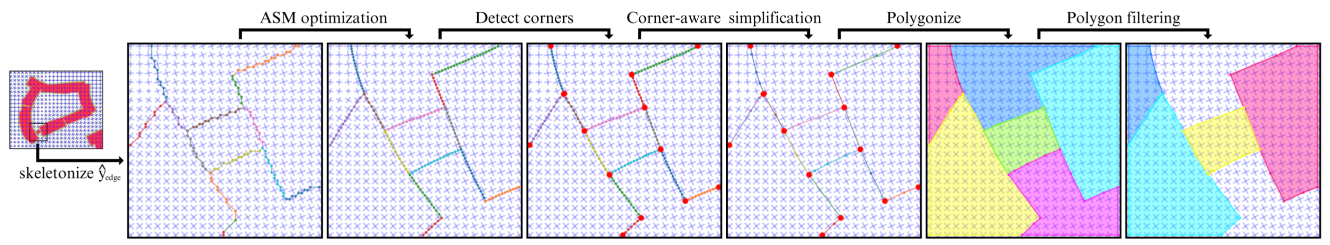


FIGURE 3

Visualization of the vectorization process [1].

- $E_{probability}$ – fits the skeleton paths to the contour of the building interior probability map at a certain probability threshold, e.g. 0.5
- $E_{frame\ field\ align}$ – aligns each edge of the skeleton graph to the frame field.
- E_{length} – ensures that the node distribution along paths remains homogeneous as well as tight.

UNIMATCH SEMI-SUPERVISED LEARNING

UniMatch [2], an advanced semi-supervised learning method in the consistency regularization category, builds upon the foundational principles established by FixMatch [3], a baseline method in this domain. primarily operates on the principle of pseudo-labeling combined with consistency regularization.

The basic principle of the FixMatch method involves generating pseudo-labels for unlabeled data from the predictions of a neural network. Specifically, for a weakly perturbed unlabeled input x^w, a prediction p^w is generated, which serves as a pseudo-label for the prediction of x^s, a strongly perturbed input. Subsequently, the loss function value, for example, cross-entropy(p^w, p^s), is calculated, considering only areas from pw with a probability value greater than a certain threshold, e.g., >0.95.

UniMatch builds upon and extends the FixMatch methodology, introducing two core enhancements:

Rozšírenie metódy UniMatch oproti metóde FixMatch spočíva v dvoch princípoch:

1. UniPerb (Unified Perturbations for Images and Features) – This involves applying perturbations at the feature level. Practically, this means applying a dropout function to the output (i.e., the feature) from the encoder layer of the neural network, randomly ignore features, which then proceed to the decoder part of the network, generating p^{fp}.
2. DusPerb (Dual-Stream Perturbations) – namiesto jednej silnej perturbácie sa využívajú dve silné perturbácie x^{s_1} a x^{s_2}.

Ultimately, there are three error functions: crossentropy(p^w, p^{fp}), crossentropy(p^w, p^{s_1}), crossentropy(p^w, p^{s_2}). These are then linearly combined with the supervised error function.

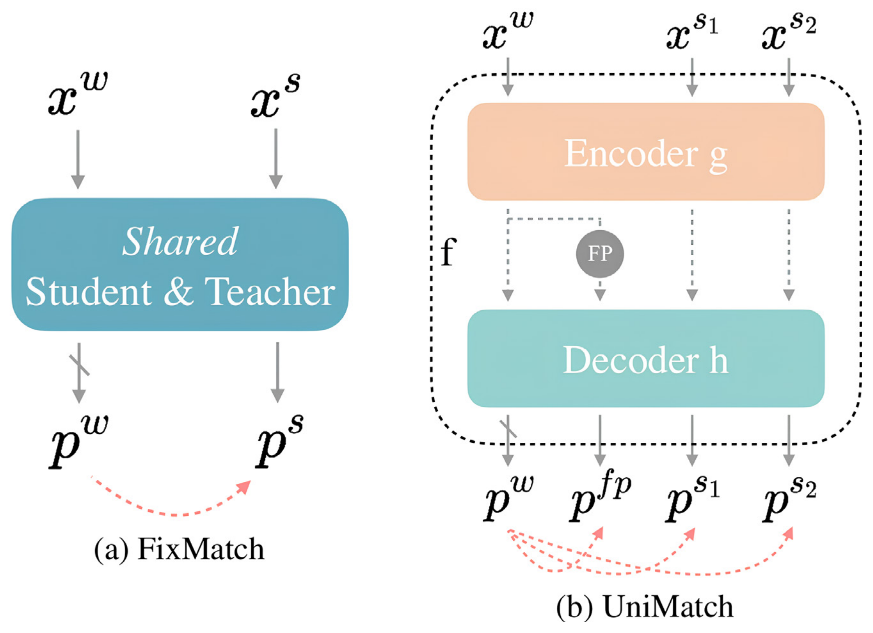


FIGURE 4

(a) The FixMatch baseline (b) used UniMatch method. The FP denotes feature pertubation, *w* and *s* means weak and strong pertubation, respectively [2].

Integrating UniMatch Semi-Supervised Learning with Frame Field Learning

IMPLEMENTATION STRATEGY FOR UNIMATCH IN FRAME FIELD LEARNING

To integrate UniMatch into our Frame Field learning framework, we first differentiated between weak and strong perturbations. For weak perturbations, we chose basic spatial transformations such as rotation, mirroring, and vertical/horizontal flips. These are well-suited for aerial imagery and straightforward to implement.

For strong perturbations, we opted for photometric transformations. These include adjustments in hue, color, and brightness, providing a more significant alteration to the images compared to spatial transformations.

Incorporating feature perturbation loss was a crucial step. We implemented this by introducing a dropout mechanism between the encoder and decoder parts of the network. This dropout selectively omits features at the feature level, which is essential for the UniMatch approach.

Regarding the dual-stream perturbations of UniMatch, we adapted our model to handle two types of strong perturbations. The dual-stream approach involves using the weak perturbation prediction as a pseudo-label and training the model using the strong perturbation predictions as loss functions. We have two strong perturbations, hence the term 'dual-stream'. Each of these perturbations contributes to the overall robustness and effectiveness of the model in semi-supervised learning scenarios, especially in the context of building extraction from complex aerial imagery.

Through these specific implementations, UniMatch was successfully integrated into the Frame Field learning model, enhancing

Experiments

DATASET

▸ *Labeled Data*

Our labeled data comes from three different sources, which we'll detail in the accompanying Table 1.

▸ *Unlabeled Data*

For the unlabeled dataset, we selected high-quality aerial images from Geodetický a kartografický ústav (GKÚ) [6], available for free public use. We specifically targeted a diverse area of 7000 square kilometers, ensuring a wide representation of various landscapes and urban settings.

▸ *Data Processing: Patching*

We processed both labeled and unlabeled images into patches of size 320x320 px. This patch size is specifically chosen to match the input requirements of our neural network. From the labeled data, this process resulted in approximately 55,000 patches. Similarly, from the unlabeled dataset, we obtained around 244,000 patches.

Name	Dataset Type	Area Covered [sq km]	Number of Buildings	Ortofotography Resolution [m]	Tile resolution [px]
Geodeticca-Buildings	Private	198.18	50 354	0.25	4,1
INRIA [4]	Open source	810.00	206 679	0.30	38,8
Landcover.ai [5]	Open source	216.27	12 354	0.25/0.50	9 000x9 500 / 4 200x4 700

TABLE 1

Overview of 3 data sources of labeled data used for training the models with details.

TRAINING SETUP

▸ *Model Architecture*

We designed our model using a U-Net architecture with an EfficientNet-B4 backbone. This combination provides a good balance of accuracy and efficiency, crucial for handling the complexity of our segmentation tasks. The EfficientNet-B4 backbone was specifically chosen for its optimal balance between memory usage and performance. In Frame Field learning, U-Net architecture has been shown to be highly effective, as evidenced by its strong performance in prior studies.

▸ *Training Process*

For training, we used the AdamW optimizer, which combines the advantages of Adam optimization with weight decay, aiding in better model generalization. To prevent overfitting, we implemented L2 regularization. Additionally, we used the ReduceLROnPlateau learning rate scheduler. This scheduler adjusts the learning rate based on validation loss, ensuring efficient training progress.

▸ *Semi-Supervised Learning Adjustments*

A key aspect of our training was adjusting the ratio of unlabeled to labeled patches. We experimented with ratios ranging from 1:1 to 1:5 (labeled:unlabeled). This variability allowed us to explore the impact of different amounts of unlabeled data on the learning process. It enabled us to identify the optimal balance for training our model, ensuring effective learning while leveraging the advantages of semi-supervised learning in handling large and diverse datasets.

Integrating extracted building data with other geospatial layers enhances the understanding of urban dynamics and spatial relationships.

MODEL EVALUATION

In our evaluation of the building footprint extraction model, we chose metrics that precisely measure how well our predictions align with real-world structures.

▸ *Intersection over Union (IoU)*

Intersection over Union (IoU) is a key metric we used. It calculates the overlap between our model's predictions and the actual building shapes. An IoU score close to 1 means our predictions closely match the real buildings. This metric is essential for assessing the geometric accuracy of the segmented areas, directly reflecting the precision of boundary delineation. Moreover, by evaluating the ratio of correctly predicted area to the combined area of prediction and ground truth, IoU provides a clear measure of the model's effectiveness in capturing the true extent and shape of buildings in complex urban landscapes.

▸ *Precision, Recall and F1*

Precision measures the accuracy of the model's building predictions, indicating the proportion of correctly identified buildings out of all identified buildings, thereby reflecting the model's specificity. Recall assesses the model's ability to capture all actual buildings, with a high recall score highlighting its sensitivity in detecting buildings. The F1 Score combines precision and recall into a single metric, offering a balanced view of the model's performance by ensuring that high scores result from both high precision and high recall.

▸ *Complexity Aware IoU (cIoU)*

We also utilized Complexity Aware IoU (cIoU) [7]. This metric addresses a shortfall in IoU by balancing segmentation accuracy and the complexity of the polygon shapes. While IoU alone can lead models to create overly complex polygons, cIoU ensures that the complexity of the polygons (number of vertices) is kept realistic, reflecting the typically less complex structure of real buildings.

▸ *N Ratio Metric*

The N ratio metric was an additional component of our evaluation strategy. It contrasts the number of vertices in our predicted shapes with those in the actual buildings [7]. This

helps in understanding whether our model accurately replicates the detailed structure of the buildings.

▸ *Max Tangent Angle Error*

To ensure clean geometry in building extraction tasks, accurately measuring contour regularity is essential. The Max Tangent Angle Error (MTAE) [1] metric is designed to address this need by supplementing the Intersection over Union (IoU) metric. It specifically targets the limitation of IoU, where segmentations with rounded corners may receive higher scores than those with more precise, sharp corners. By evaluating the alignment of edges through the comparison of tangent angles at sampled points along predicted and ground truth contours, MTAE effectively penalizes inaccuracies in edge orientation. This focus on edge precision is critical for producing clean vector representations of buildings, emphasizing the importance of accurate edge delineation in segmentation tasks.

▸ *Evaluation Process*

Trained models were tested on large, full-size aerial images, instead of small patches. This method ensured that our evaluation closely mirrored real-world scenarios. To extract buildings from full-size images, we employed a sliding window technique, aligning with the neural network's training on specific patch sizes, to systematically produce segment-wise predictions. An advanced averaging technique was applied at the borders of these overlapping segments, crucial for minimizing artifacts and ensuring consistency across the prediction map. The resulting seamless, full-size output was then vectorized into precise vector polygons using the Active Skeleton Model (ASM) algorithm.

RESULTS

The results from our experiments, reflecting performance of segmentation model trained under different conditions, reveal significant insights (see Table 2). We evaluated the model's performance in a baseline scenario without semi-supervised learning and in scenarios where semi-supervised learning was applied with varying ratios of labeled to unlabeled data (1:1, 1:3, and 1:5).

Training Approach	Ratio (Labeled: Unlabeled)	IoU [%]	Precision [%]	Recall [%]	F1 Score [%]	N Ratio	cIoU [%]	Mean MTAE [°]
Baseline	-	80.50	85.75	94.27	89.81	2.33	48.89	18.60
Semi-Supervised	1:1	83.88	87.66	93.41	90.44	1.94	56.98	20.47
Semi-Supervised	1:3	85.35	90.04	94.25	92.10	1.76	61.91	18.92
Semi-Supervised	1:5	85.77	90.04	94.76	92.34	1.65	64.75	17.45

TABLE 2

Results of the models' training for baseline (supervised) a semi-supervised approaches with different ratios of labeled to unlabeled images used.

1. IoU Percentage Increase:
   Starting from the baseline IoU of 80.50%, we observed a steady increase in this metric as we introduced more unlabeled data into the training process, reaching up to 85.77% with a 1:5 labeled to unlabeled ratio.
2. Precision, Recall, and F1 Score:
   The precision of the model, which measures how accurate the predictions are, improved from 85.75% in the baseline to 90.04% in the 1:5 ratio setup. Similarly, recall, which indicates how well the model can find all relevant instances, slightly increased from 94.27% to 94.76%. The F1 Score, which balances precision and recall, also saw an improvement from 89.81% to 92.34%. These improvements suggest that the model became more accurate and reliable in its predictions when semi-supervised learning was used.
3. N Ratio and cIoU:
   The results show a notable decrease in the N Ratio from 2.33 in the baseline to 1.65 in the semi-supervised 1:5 ratio setup, indicating that the semi-supervised model generates simpler, yet accurate, vector shapes that more closely resemble the actual structures. This simplification likely contributes to the enhanced usability of the output in practical GIS applications. Concurrently, the complexity-aware IoU (cIoU) significantly improved from 48.89% in the baseline to 64.75% in the 1:5 ratio, suggesting that the semi-supervised learning approach not only improves the overlap between the predicted and actual building footprints but also produces simpler vector shapes, which are closer to real-world buildings in terms of geometry.
4. Pean Max Tangent Angle Error (MTAE):
   The Mean MTAE's reduction from 18.60° in the baseline to 17.45° in the 1:5 semi-supervised setting signifies an improvement in the geometric precision of the model's predictions. This suggests that the semi-supervised learning model is better at capturing the architectural features of buildings with more accurately defined angles, contributing to the production of topologically simpler and cleaner vector polygons.

Training on High-Performance Computing (HPC) Machine

HPC CONFIGURATION

Our training was conducted on a High-Performance Computing (HPC) machine equipped with substantial computational resources. The HPC had 8 nodes, each outfitted with 4 NVIDIA A100 GPUs with 40GB of VRAM, 64 CPU cores, and 256GB of RAM. For task scheduling, the system utilized Slurm.

PYTORCH LIGHTNING FRAMEWORK

We employed the PyTorch Lightning framework, which offers user-friendly multi-GPU settings. This framework allows the specification of the number of GPUs per node, the total number of nodes, various distributed strategies, and the option for mixed-precision training.

EXPERIENCES WITH SLURM AND PYTORCH LIGHTNING

When training on a single GPU, our Slurm configuration was as follows:

```
#SBATCH --partition=ngpu
#SBATCH --gres=gpu:1
#SBATCH --cpus-per-task=16
#SBATCH –mem=64000
```

In PyTorch Lightning, we set the trainer as:

```
trainer = Trainer(accelerator="gpu", devices=1)
```

Since, here, we allocated one GPU from four available in one node, we allocated 16 CPUs from 64 available. Therefore, for the data loaders, we assigned 16 workers. Since semi-supervised learning uses two data loaders (one for labeled and one for unlabeled data), we allocated 8 workers to each. It was critical to ensure that the total number of cores for the data loaders did not exceed the available CPUs to prevent training crashes.

DISTRIBUTED DATA PARALLEL (DDP) STRATEGY

Using PyTorch Lightning's Distributed Data Parallel (DDP) option, we ensured each GPU across the nodes operated independently:

- Each GPU processed a portion of the dataset.
- All processes initiated the model independently.
- Each conducted forward and backward passes in parallel.
- Gradients were synchronized and averaged across processes.
- Each process updated its optimizer individually

With this approach, the total number of data loaders equaled the number of GPUs multiplied by the number of data loaders. For example, in a semi-supervised learning setup with 4 GPUs and two types of data loaders (labeled and unlabeled), we ended up with 8 data loaders, each with 8 workers – 64 workers in total.

To fully utilized one node with four GPU, we used following configurations:

```
#SBATCH --partition=ngpu
#SBATCH --gres=gpu:4
#SBATCH –exclusive
#SBATCH --cpus-per-task=64
#SBATCH –mem=256000
```

In PyTorch Lightning, we set the trainer as:

```
trainer = Trainer(accelerator="gpu", devices=4, strategy="ddp")
```

UTILIZING MULTIPLE NODES

Using PyTorch Lighting, it is possible to leverage multiple nodes on HPC. For instance, using 4 nodes with 4 GPUs each (16 GPUs in total) was configured as:

```
trainer = Trainer(accelerator="gpu", devices=4, strategy="ddp", num_nodes=4)
```

Correspondingly, the Slurm configuration was set to:

```
#SBATCH –nodes=4
#SBATCH –ntasks-per-node=4
#SBATCH --gres=gpu:4
```

These settings and experiences highlight the scalability and flexibility of training complex machine learning models on an HPC environment, especially for tasks demanding significant computational resources like semi-supervised learning in geospatial data analysis.

TRAINING SCALABILITY ANALYSIS

In the Training Scalability Analysis, we carefully examined the impact of expanding computational resources on the efficiency of training models, utilizing the PyTorch Lightning framework. This investigation covered both supervised and semi-supervised learning approaches, with a particular emphasis on the effects of increasing GPU numbers, including setups involving 2 nodes (or 8 GPUs).

A key finding from this analysis was that the increase in speedup ratios for supervised learning did not perfectly align with the number of GPUs utilized. Ideally, doubling the number of GPUs would directly double the speedup ratio (e.g., using 4 GPUs would result in a 4x speedup). However, the actual speedup ratios were lower than this ideal expectation. This discrepancy can be attributed to the overhead associated with managing multiple GPUs and nodes, particularly the

Training type	# GPU	Time per epoch [min]	Speedup Ratio [vs. 1 GPU]
Supervised learning	8 (2 nodes)	1:25	5.56x
	4	2:38	2.99x
	2	4:01	1.96x
	1	7:53	1.00x
Semi-supervised learning with ratio 1:1	8 (2 nodes)	3:55	7.25x
	4	7:17	3.90x
	2	14:24	1.97x
	1	28:23	1.00x

TABLE 3

Training results of supervised and semi-supervised approaches with 1, 2, and 4 GPUs. Time for one epoch is reported for each configuration.

need to synchronize data across all GPUs, which introduces efficiency losses.

In contrast, semi-supervised learning exhibited a trend that more closely approached the ideal linear increase in speedup ratios. The complex nature and higher computational requirements of semi-supervised learning tasks seem to diminish the relative effect of overhead costs, thereby allowing for a more efficient use of additional GPUs. Despite the challenges of data synchronization across multiple GPUs and nodes, the intensive computational demands of semi-supervised learning enable a more effective scaling of resources, yielding speedup ratios that more closely mirror the ideal scenario.

Conclusion

The research presented in this whitepaper has successfully demonstrated the effectiveness of integrating UniMatch semi-supervised learning with Frame Field learning for the task of building extraction from aerial imagery. This integration addresses the challenges associated with the scarcity of labeled data in deep learning applications for geographic information systems (GIS), providing a cost-effective and scalable solution.

Our findings reveal that employing semi-supervised learning significantly enhances the model's performance across several key metrics, including Intersection over Union (IoU), precision, recall, F1 Score, N Ratio, complexity-aware IoU (cIoU), and Mean Max Tangent Angle Error (MTAE). Notably, the improvements in IoU and cIoU metrics underscore the model's increased accuracy in delineating building footprints and generating vector shapes that closely resemble actual structures. This outcome is pivotal for applications in urban planning, environmental studies, and infrastructure management, where precise mapping and analysis of building data are crucial.

The methodology adopted, which combines Frame Field learning with the innovative UniMatch approach, has proven to be highly effective in leveraging both labeled and unlabeled data. This strategy not only improves the geometric precision of the model's predictions but also ensures the generation of cleaner, topologically accurate vector polygons. Furthermore, the scalability and efficiency of training on a High-Performance Computing (HPC) machine using the PyTorch Lightning framework and Distributed Data Parallel (DDP) strategy have been instrumental in handling the extensive computational demands of the semi-supervised learning process on the data at hand, within a time frame ranging from tens of minutes to hours.

In conclusion, this research highlights the potential of semi-supervised learning in significantly advancing the field of

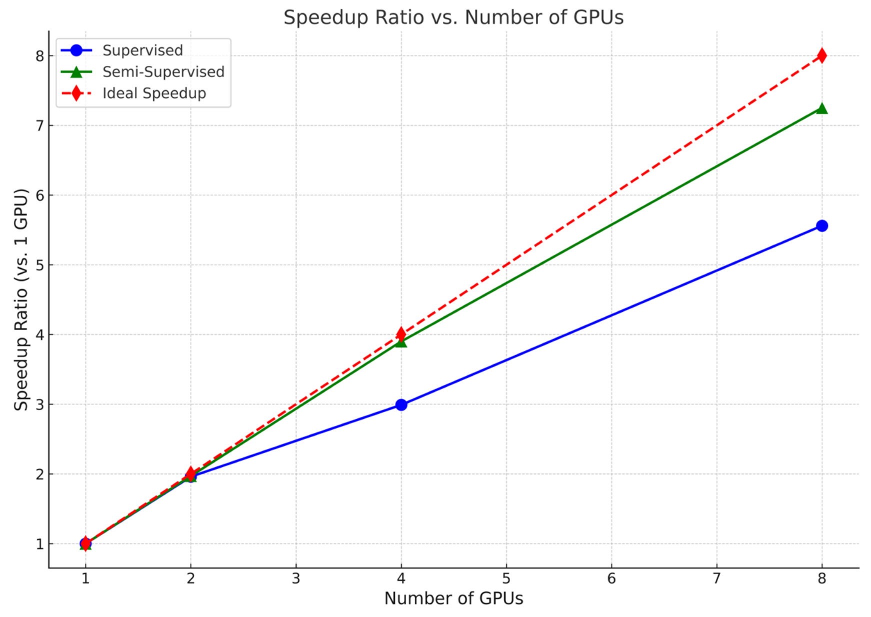
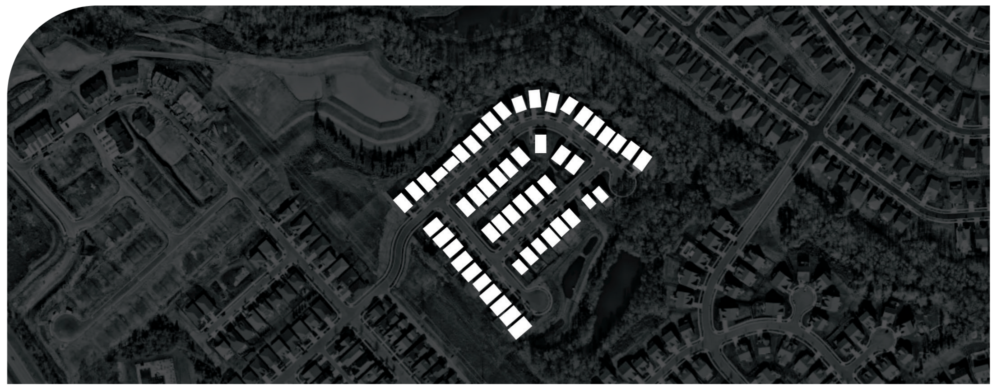


FIGURE 5

This graph compares the actual speedup ratios for supervised and semi-supervised learning against the number of GPUs, alongside the ideal linear speedup ratio. It showcases the closer alignment of semi-supervised learning with ideal scalability, emphasizing its greater efficiency gains from increased computational resources.

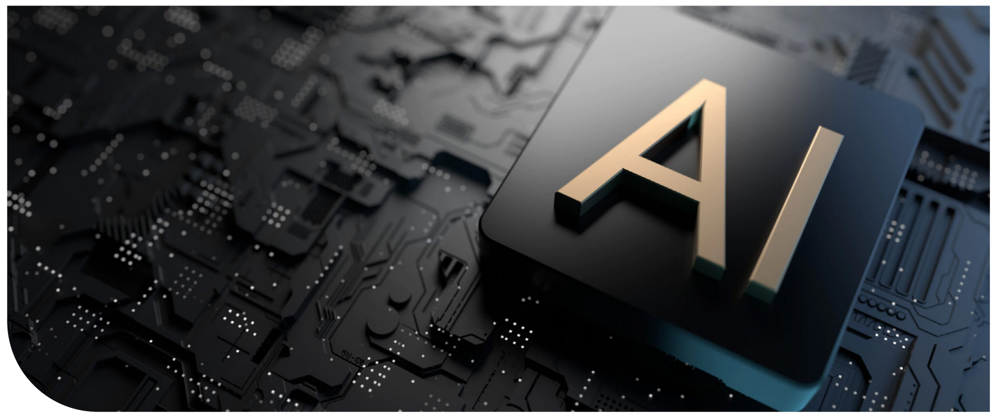


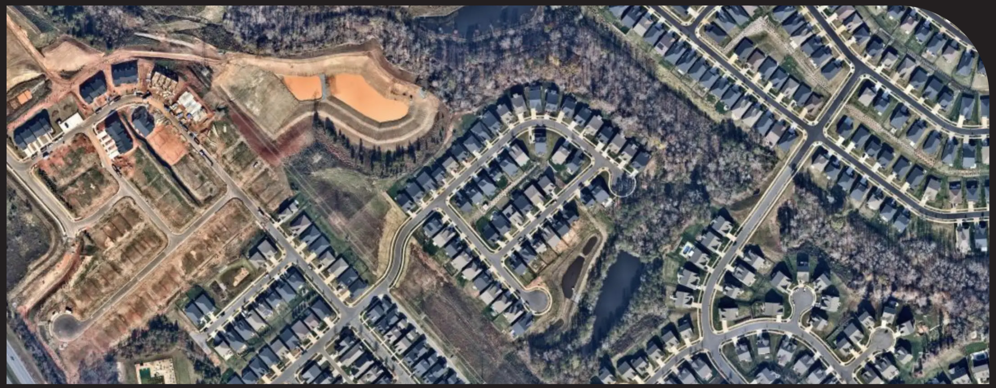
automated building extraction from aerial imagery. The implementation of UniMatch within the Frame Field learning framework represents a notable leap forward, offering a robust solution to the challenges of data scarcity and the need for high-precision geospatial data analysis. This approach not only enhances the efficiency and accuracy of building extraction but also opens new avenues for the application of semi-supervised learning techniques in GIS and related fields.

ACKNOWLEDGEMENTS

Research results were obtained with the support of the Slovak National competence centre for HPC, the EuroCC 2 project and Slovak National Supercomputing Centre under grant agreement 101101903-EuroCC 2-DIGITAL-EUROHPC-JU-2022-NCC-01.

Computational resources were procured in the national project National competence centre for high performance computing (project code: 311070AKF2) funded by European Regional Development Fund, EU Structural Funds Informatization of society, Operational Program Integrated Infrastructure.





REFERENCES

[1] Nicolas Girard, Dmitriy Smirnov, Justin Solomon, and Yuliya Tarabalka. Polygonal Building Extraction by Frame Field Learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (June 2021), pp. 5891-5900.

[2] L. Yang, L. Qi, L. Feng, W. Zhang, and Y. Shi. *Revisiting Weak-to-Strong Consistency in Semi-Supervised Semantic Segmentation.* In: 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (June 2023), pp. 7236-7246. doi: 10.1109/CVPR52729.2023.00699.

[3] Kihyuk Sohn, David Berthelot, Chun-Liang Li, Zizhao Zhang, Nicholas Carlini, Ekin D. Cubuk, Alex Kurakin, Han Zhang, and Colin Raffel. FixMatch: Simplifying Semi-Supervised Learning with Consistency and Confidence. In *CoRR*, vol. abs/2001.07685 (2020). Available: https://arxiv.org/abs/2001.07685.

[4] Emmanuel Maggiori, Yuliya Tarabalka, Guillaume Charpiat, and Pierre Alliez. Can Semantic Labeling Methods Generalize to Any City? The Inria Aerial Image Labeling Benchmark. In *IEEE International Geoscience and Remote Sensing Symposium (IGARSS)* (2017). IEEE.

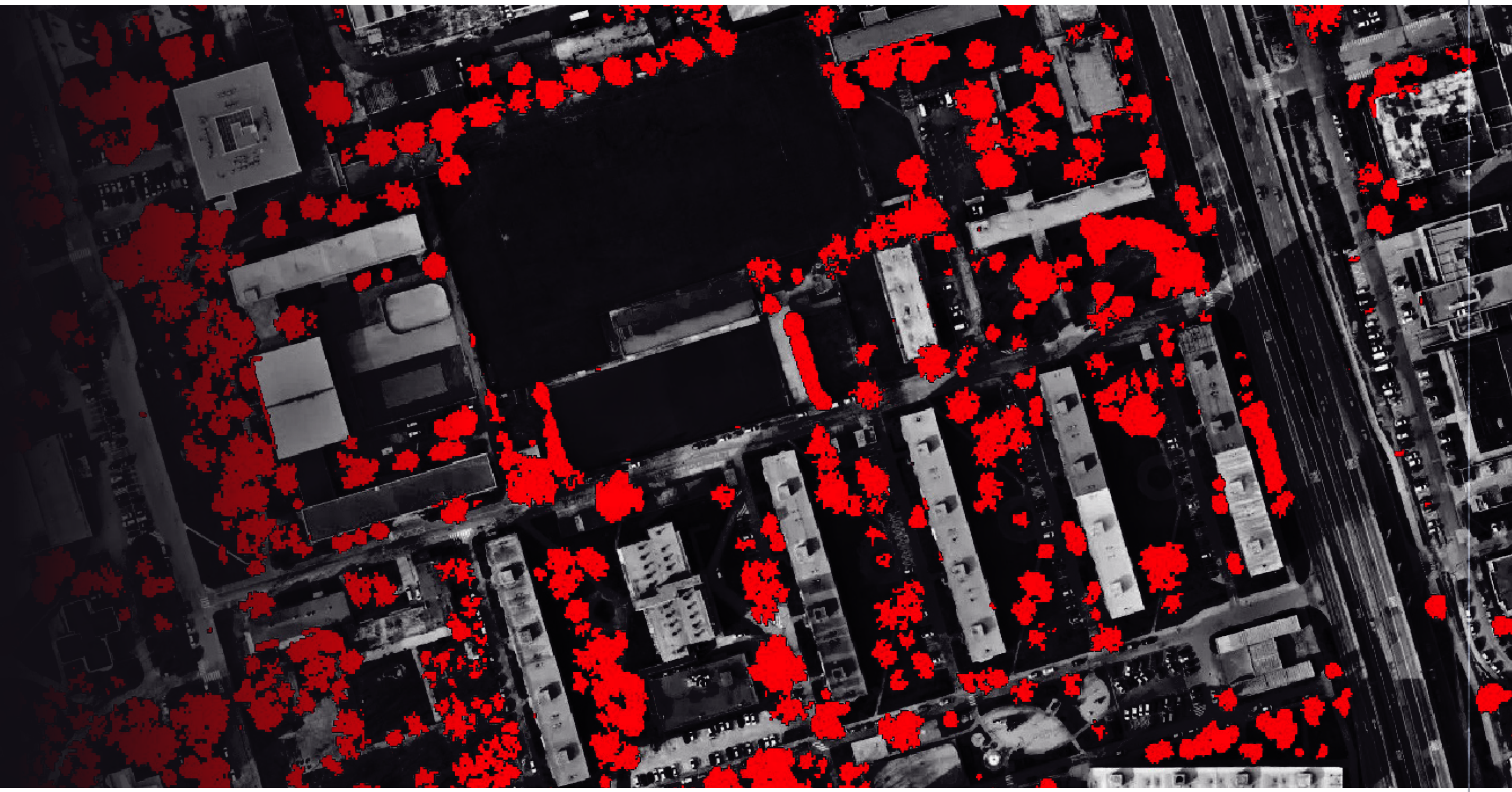
[5] Adrian Boguszewski, Dominik Batorski, Natalia Ziemba-Jankowska, Tomasz Dziedzic, and Anna Zambrzycka. LandCover.ai: Dataset for Automatic Mapping of Buildings, Woodlands, Water and Roads from Aerial Imagery. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops* (June 2021), pp. 1102-1110.

[6] Ortofotomozaika. *Geoportal SK*. Accessed February 14, 2024. https://www.geoportal.sk/sk/zbgis/ortofotomozaika/.

[7] Stefano Zorzi, Shabab Bazrafkan, Stefan Habenschuss, and Friedrich Fraundorfer. PolyWorld: Polygonal Building Extraction with Graph Neural Networks in Satellite Images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2022), pp. 1848-1857.

Mapping tree positions and heights using pointCloud data obtained using LiDAR Technology

Marián Gall, Michal Malček
Dávid Murín, Robert Straka



The goal of the collaboration between the Slovak National SuperComputing Center (NSCC) and the company SKYMOVE, within the National Competence Center for HPC project was to design and implement a pilot software solution for processing data obtained using LiDAR (Light Detection and Ranging) technology mounted on drones.

LiDAR is an innovative method of remote distance measurement that is based on measuring the travel time of laser pulse reflections from objects. LiDAR emits light pulses that hit the ground or object and return to the sensors. By measuring the return time of the light, LiDAR determines the distance to the point where the laser beam was reflected.

LiDAR can emit 100k to 300k pulses per second, capturing dozens to hundreds of pulses per square meter of the surface, depending on specific settings and the distance to the scanned object. This process creates a point cloud (PointCloud) consisting of potentially millions of points. Modern LiDAR use involves data collection from the air, where the device is mounted on a drone, increasing the efficiency and accuracy of data collection. In this project, drones from DJI, particularly the DJI M300 and Mavic 3 Enterprise (Fig. 1), were used for data collection. The DJI M300 is a professional drone designed for various industrial applications, and its parameters make it suitable for carrying LiDAR.

The DJI M300 drone was used as a carrier for the Geosun LiDAR (Fig. 1). This is a mid-range, compact system with an integrated laser scanner, a positioning, and orientation system. Given the balance between data collection speed and data quality, the data was scanned from a height of 100 meters above the surface, allowing for the scanning of larger areas in a relatively short time with sufficient quality.

The collected data was geolocated in the S-JTSK coordinate system (EPSG:5514) and the Baltic Height System after adjustment (Bpv), with coordinates given in meters or meters above sea level. In addition to LiDAR data, aerial photogrammetry was performed simultaneously, allowing for the creation of orthophotomosaics. Orthophotomosaics provide a photographic record of the surveyed area in high resolution (3 cm/pixel) with positional accuracy up to 5 cm. The orthophotomosaic was used as a basis for visual verification of the positions of individual trees.

LiDAR is an innovative method of remote distance measurement that is based on measuring the travel time of laser pulse reflections from objects.

Data classification

The primary dataset used for the automatic identification of trees was a LiDAR point cloud in LAS/LAZ format (uncompressed and compressed form). LAS files are a standardized format for storing LiDAR data, designed to ensure efficient storage of large amounts of point data with precise 3D coordinates. LAS files contain information about position (x, y, z), reflection intensity, point classification, and other attributes necessary for LiDAR data analysis and processing. Due to their standardization and compactness, LAS files are widely used in geodesy,

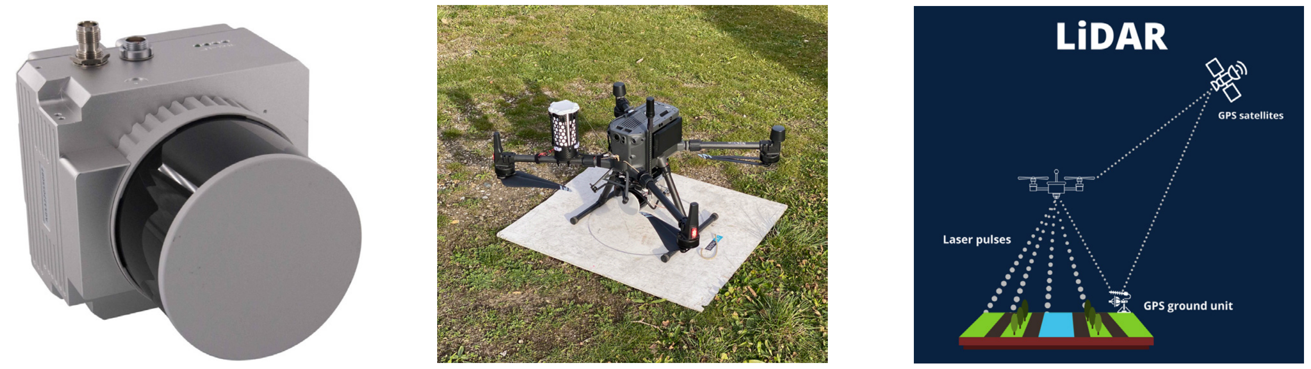


FIGURE 1

Geosun LiDAR (left), DJI M300 drone (center) and remote sensing scheme (right).

cartography, forestry, urban planning, and many other fields requiring detailed and accurate 3D representations of terrain and objects.

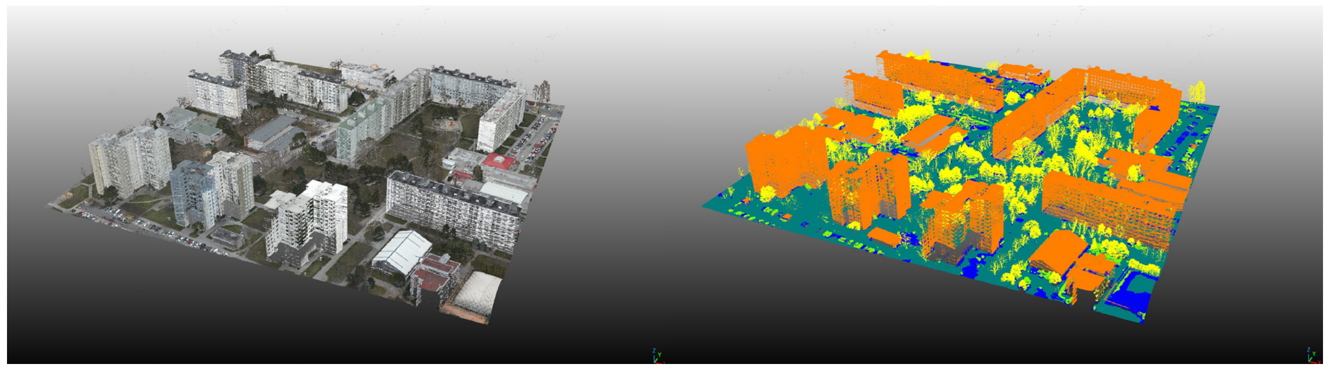
The point cloud needed to be processed into a form that would allow for an easy identification of individual tree or vegetation points. This process involves assigning a specific class to each point in the point cloud, known as classification.

Various tools can be used for point cloud classification. Given our positive experience, we decided to use the Lidar360 software from GreenValley International [1]. In the point cloud classification, the individual points were classified into the following categories: unclassified (1), ground (2), medium vegetation (4), high vegetation (5), buildings (6). A machine learning method was used for classification, which, after being trained on a representative training sample, can automatically classify points of any input dataset (Fig. 2).

The training sample was created by manually classifying points in the point cloud into the respective categories. For the purposes of automated tree identification in this project, the ground and high vegetation categories are essential. However, for the best classification results of high vegetation, it is also advisable to include other classification categories. The training sample was composed of multiple smaller areas from the entire region including all types of vegetation, both deciduous and coniferous, as well as various types of buildings. Based on the created training sample, the remaining points of the point cloud were automatically classified. It should be noted that the quality of the training sample significantly affects the final classification of the entire area.

FIGURE 2

Example of a point cloud of an area colored using an orthophotomosaic (left) and the corresponding classification (right) in CloudCompare.

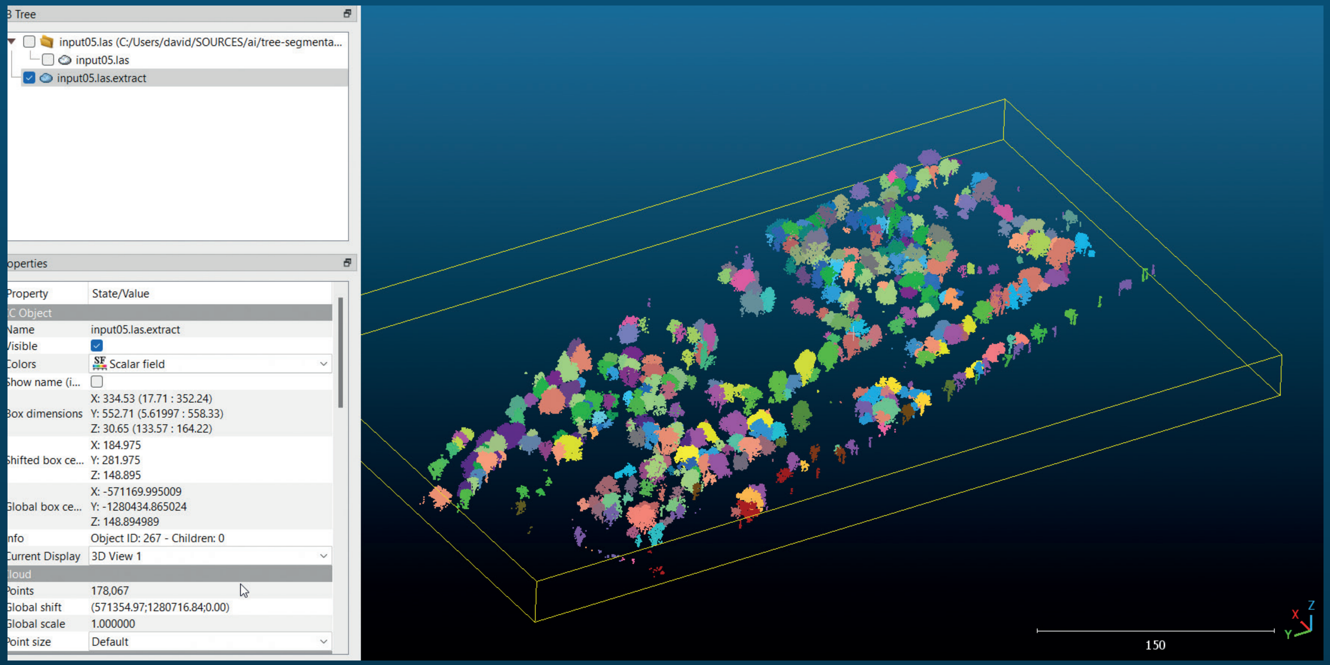


Data segmentation

In the next step, the classified point cloud was segmented using the CloudCompare software [2]. Segmentation generally means dividing classified data into smaller units – segments that share common characteristics. The goal of segmenting high vegetation was to assign individual points to specific trees.

For tree segmentation, the TreeIso plugin in the CloudCompare software package was used, which automatically recognizes trees based on various height and positional criteria (Fig. 3). The overall segmentation consists of three steps:

1. Grouping points that are close together into segments and removing noise.
2. Merging neighboring point segments into larger units.
3. Composing individual segments into a whole that forms a single tree.



3 Tree

input05.las (C:/Users/david/SOURCES/ai/tree-segmenta...
input05.las
input05.las.extract

operties

Property	State/Value
C Object	
Name	input05.las.extract
Visible	✓
Colors	Scalar field
Show name (i...	☐
Box dimensions	X: 334.53 (17.71 : 352.24) Y: 552.71 (5.61997 : 558.33) Z: 30.65 (133.57 : 164.22)
Shifted box ce...	X: 184.975 Y: 281.975 Z: 148.895
Global box ce...	X: -571169.995009 Y: -1280434.865024 Z: 148.894989
Info	Object ID: 267 - Children: 0
Current Display	3D View 1
Cloud	
Points	178,067
Global shift	(571354.97;1280716.84;0.00)
Global scale	1.000000
Point size	Default

FIGURE 3

Segmented point cloud in CloudCompare using the TreeIso plugin module.

The result is a complete segmentation of high vegetation. These segments are then saved into individual LAS files and used for further processing to determine the positions of individual trees. A significant drawback of this tool is that it operates only in serial mode, meaning it can utilize only one CPU core, which greatly limits its use in an HPC environment.

As an alternative method for segmentation, we explored the use of orthophotomosaics of the studied areas. Using machine learning methods, we attempted to identify individual tree crowns in the images and, based on the geolocational coordinates determined, identify the corresponding segments in the LAS file. For detecting tree crowns from the orthophotomosaic, the YOLOv5 model [3] with pretrained weights from the COCO128 database [4] was used. The training data consisted of 230 images manually annotated using the LabelImg tool [5]. The training unit consisted of 300 epochs, with images divided into batches of 16 samples, and their size was set to 1000x1000 pixels, which proved to be a suitable compromise between computational demands and the number of trees per section. The insufficient quality of this approach was particularly evident in areas with dense vegetation (forested areas), as shown in Figure 4. We believe this was due to the insufficient robustness of the chosen training set, which could not adequately cover the diversity of image data (especially for different vegetative periods). For these reasons, we did not develop segmentation from photographic data further and focused solely on segmentation in the point cloud.

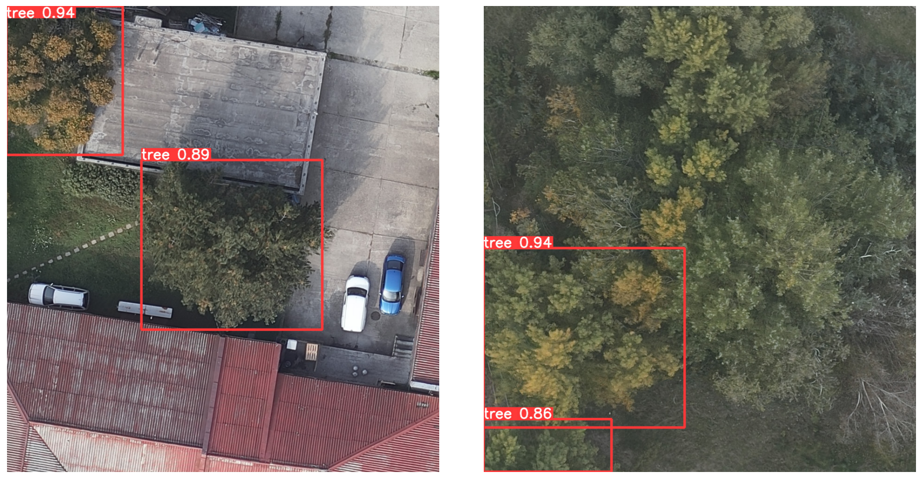


FIGURE 4

Tree segmentation in the orthophotomosaic using the YOLOv5 tool. The image illustrates the problem of detecting individual trees in the case of dense vegetation (continuous canopy).

To fully utilize the capabilities of the Devana supercomputer, we deployed the lidR library [6] in its environment. This library, written in R, is a specialized tool for processing and analyzing LiDAR data, providing an extensive set of functions and tools for reading, manipulating, visualizing, and analyzing LAS files. With lidR, tasks such as filtering, classification, segmentation, and object extraction from point clouds can be performed efficiently. The library also allows for surface interpolation, creating digital terrain models (DTM) and digital surface models (DSM), and calculating various metrics for vegetation and landscape structure. Due to its flexibility and performance, lidR is a popular tool in geoinformatics and is also suitable for HPC environments, as most of its functions and algorithms are fully parallelized within a single compute node, allowing for full utilization of available hardware. When processing large datasets where the performance or capacity of a single compute node is insufficient, splitting the dataset into smaller parts and processing them independently can leverage multiple HPC nodes simultaneously.

The lidR library includes the *locate_trees()*, function, which can reliably identify tree positions. Based on selected parameters and algorithms, the function analyzes the point cloud and identifies tree locations. In our case, the *lmf* algorithm, based on maximum height localization, was used [7]. The algorithm is fully parallelized, enabling efficient processing of relatively large areas in a short time.

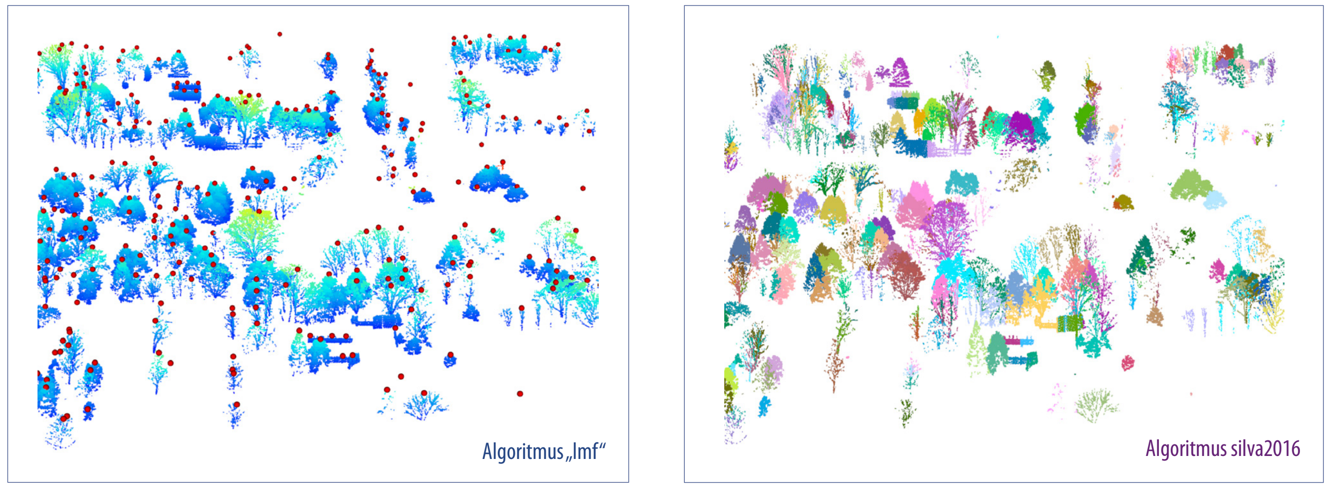


FIGURE 5

Tree positions determined using the *lmf* algorithm (left, red dots) and corresponding tree segments identified by the silva2016 algorithm (right) using the lidR library.

The identified tree positions can then be used in the silva2016 algorithm for segmentation with the *segment_trees()* function [8]. This function segments the identified trees into separate LAS files (Fig. 5), similar to the TreeIso plugin module in CloudCompare. These segmented trees in LAS files are then used for further processing, such as determining the positions of individual trees using the DBSCAN clustering algorithm [9].

Detection of tree trunks using the DBSCAN clustering algorithm

To determine the position and height of trees in individual LAS files obtained from segmentation, we used various approaches. The height of each tree was obtained based on the z-coordinates for each LAS file as the difference between the minimum and maximum coordinates of the point clouds. Since some point cloud segments contained more than one tree, it was necessary to identify the number of tree trunks within these segments.

Tree trunks were identified using the DBSCAN clustering algorithm with the following settings: maximum distance between two points within one cluster (= 1 meter) and minimum number of points in one cluster (= 10). The position of each identified trunk was then obtained based on the x and y coordinates of the cluster centroids. The identification of clusters using the DBSCAN algorithm is illustrated in Figure 6.

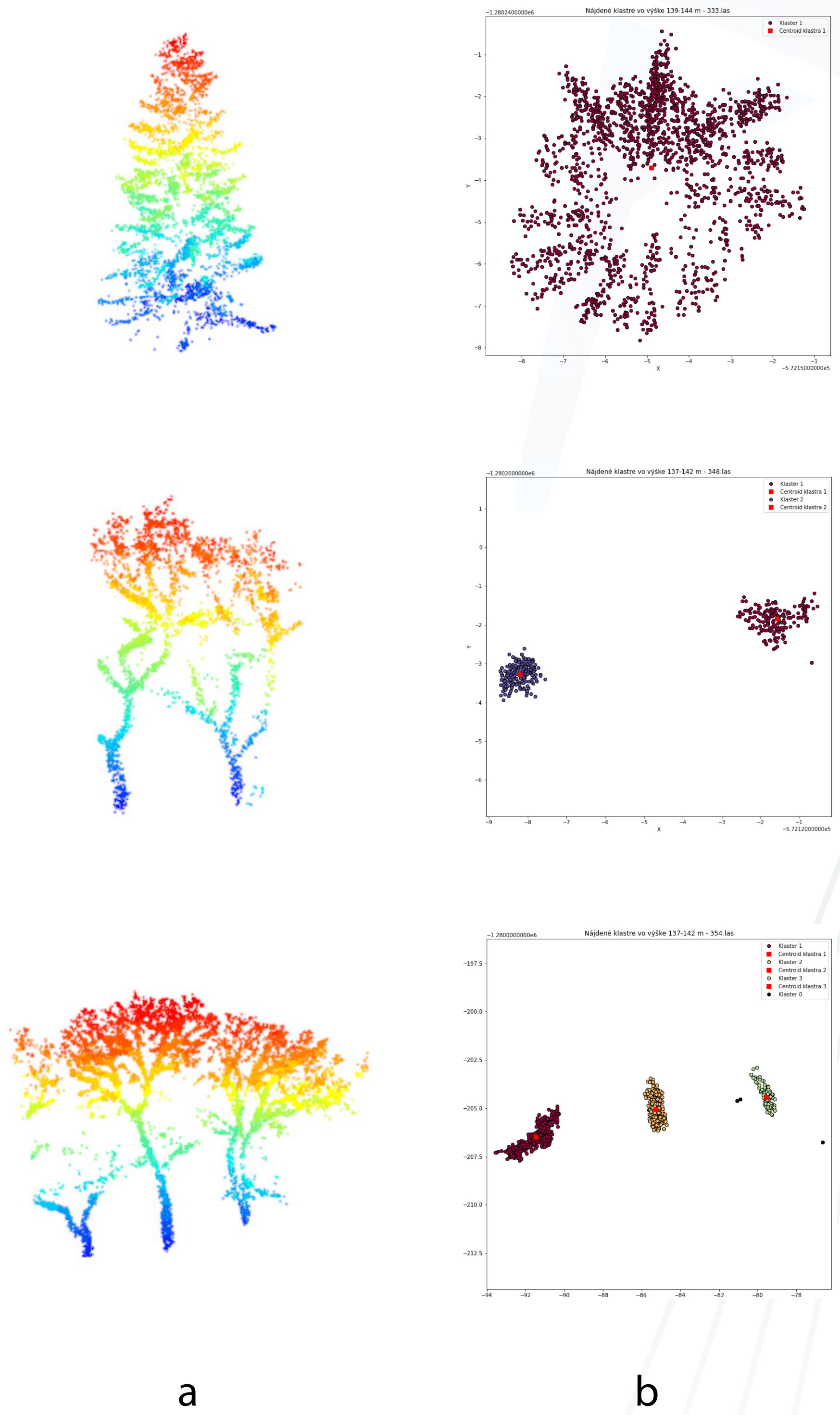


FIGURE 6

Segments of the point cloud, PointCloud (left column), and the corresponding detected clusters at heights of 1-5 meters (right column).

Determining tree heights using surface interpolation

As an alternative method for determining tree heights, we used the **Canopy Height Model** (CHM). CHM is a digital model that represents the height of the tree canopy above the terrain. This model is used to calculate the height of trees in forests or other vegetative areas. CHM is created by subtracting the **Digital Terrain Model** (DTM) from the **Digital Surface Model** (DSM). The result is a point cloud, or raster, that shows the height of trees above the terrain surface (Fig. 7).

If the coordinates of tree's position are known, we can easily determine the corresponding height of the tree at that point using this model. The calculation of this model can be easily performed using the lidR library with the *grid_terrain()* function, which creates the DTM, and the *grid_canopy()* function, which calculates the DSM.

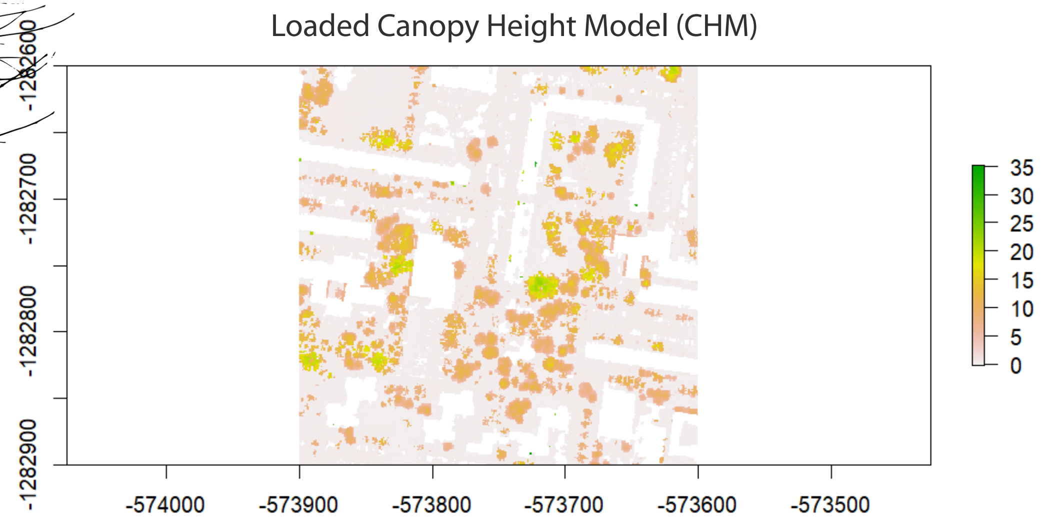


FIGURE 7

Canopy Height Model (CHM) for the studied area (coordinates in meters on the X and Y axes), with the height of each point in meters represented using a color scale.

Comparison of results

To compare the results achieved by the approaches mentioned before, we focused on the Petržalka area in Bratislava, where manual measurements of tree positions and heights had already been conducted. From the entire area (approximately 3500x3500 m), we selected a representative smaller area of 300x300 m (Fig. 2). We obtained results for the TreeIso plugin module in CloudCompare (CC), working on a PC in a Windows environment, and results for the *locate_trees()* and *segment_trees()* algorithms using the lidR library in the HPC environment of the Devana supercomputer.

We qualitatively and quantitatively evaluated the tree positions using the Munkres (Hungarian Algorithm) [10] for optimal matching. The Munkres algorithm, also known as the Hungarian Algorithm, is an efficient method for finding the optimal matching in bipartite graphs. Its use in matching trees with manually determined positions means finding the best match between trees identified from LiDAR data and their known positions. By setting an appropriate distance threshold in meters (e.g., 5 m), we can qualitatively determine the number of accurately identified tree positions. The results are processed using histograms and percentage accuracy of tree positions depending on the chosen precision threshold (Fig. 8).

We found that both methods achieve almost the same result at a 5-meter distance threshold, approximately 70% accurate tree positions. The method used in CloudCompare shows better results, i.e., a higher percentage at lower threshold values, as reflected in the corresponding histograms (Fig. 8). When comparing both methods, we achieve up to approximately 85% agreement at a threshold of up to 5 meters, indicating the qualitative parity of both approaches. The quality of the results is mainly influenced by the accuracy of vegetation classification in point clouds, as the presence of various artifacts incorrectly classified as vegetation distorts the results. Tree segmentation algorithms cannot eliminate the impact of these artifacts.

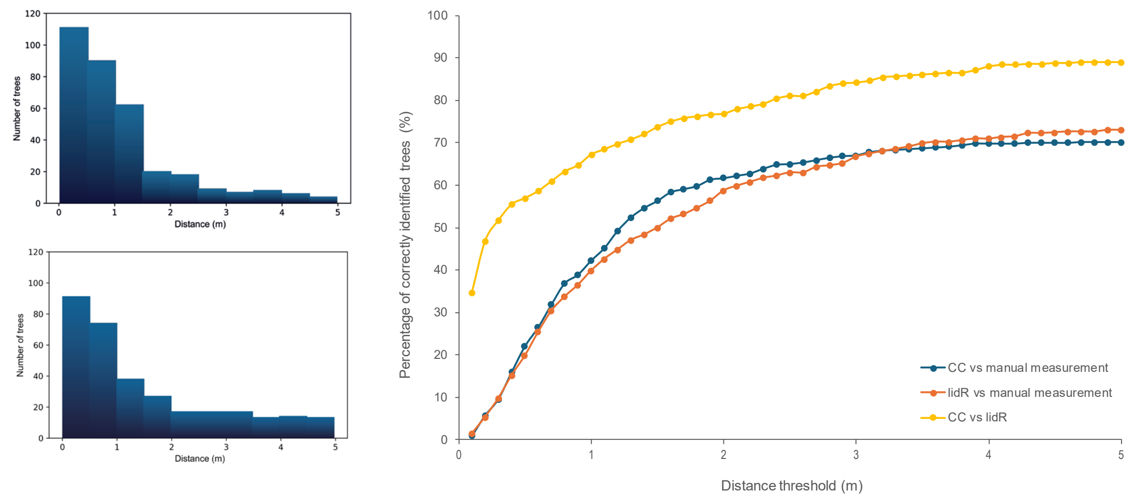


FIGURE 8

The histograms on the left display the number of correctly identified trees depending on the chosen distance threshold in meters (top: CC – CloudCompare – method, bottom: lidR method). The graphs on the right show the percentage success rate of correctly identified tree positions based on the method used and the chosen distance threshold in meters.

Parallel efficiency analysis of the locate_trees() algorithm in the lidR library

To determine the efficiency of parallelizing the *locate_trees()* algorithm in the lidR library, we applied the algorithm to the same study area using different numbers of CPU cores – 1, 2, 4, up to 64 (the maximum of the compute node of Devana HPC system). To assess sensitivity to problem size, we tested it on three areas of different sizes – 300x300, 1000x1000, and 3500x3500 meters. The times measured are shown in Table 1, and the scalability of the algorithm is illustrated in Figure 9. The results show that the scalability of the algorithm is not ideal. When using approximately 20 CPU cores, the algorithm's efficiency drops to about 50%, and with 64 CPU cores, the efficiency is only 15-20%. The efficiency is also affected by the problem size – the larger the area, the lower the efficiency, although this effect is not as pronounced. In conclusion, for effective use of the algorithm, it is suitable to use 16-32 CPU cores and to achieve maximum efficiency of the available hardware by appropriately dividing the study area into smaller parts. Using more than 32 CPU cores is not efficient but still allows for further acceleration of the computation.

TABLE 1

Achieved runtimes [s] of the *lmf* algorithm in the *locate_trees()* function of the lidR library (t, in seconds) with different number of CPU cores and varying sizes of the studied area (A, [m x m]).

A / [m x m]	CPU						
	1	2	4	8	16	32	64
300 x 300	34,4	24,0	10,7	6,8	4,1	2,6	1,5
1 000 x 1 000	363,0	187,8	110,4	60,6	38,8	29,7	22,0
3 500 x 3 500	11520,0	5832,0	3376,8	2156,4	1502,4	1029,6	822,6

Final evaluation

We found that achieving good results requires carefully setting the parameters of the algorithms used, as the number and quality of the resulting tree positions depend heavily on these settings. If obtaining the most accurate results is the goal, a possible strategy would be to select a representative part of the study area, manually determine the tree positions, and then adjust the parameters of the respective algorithms. These optimized settings can then be used for the analysis of the entire study area.

The quality of the results is also influenced by various other factors, such as the season, which affects vegetation density, the density of trees in the area, and the species diversity of the vegetation. The quality of the results is further impacted by the quality of vegetation classification in the point cloud, as the presence of various artifacts, such as parts of buildings, roads, vehicles, and other objects, can negatively affect the results. The tree segmentation algorithms cannot always reliably filter out these artifacts.

Regarding computational efficiency, we can conclude that using an HPC environment provides a significant opportunity for accelerating the evaluation process. For illustration, processing the entire study area of Petržalka (3 500 x 3 500 m) on a single compute node of the Devana HPC system took approximately 820 seconds, utilizing all 64 CPU cores. Processing the same area in CloudCompare on a powerful PC using a single CPU core took approximately 6 200 seconds, which is about 8 times slower.

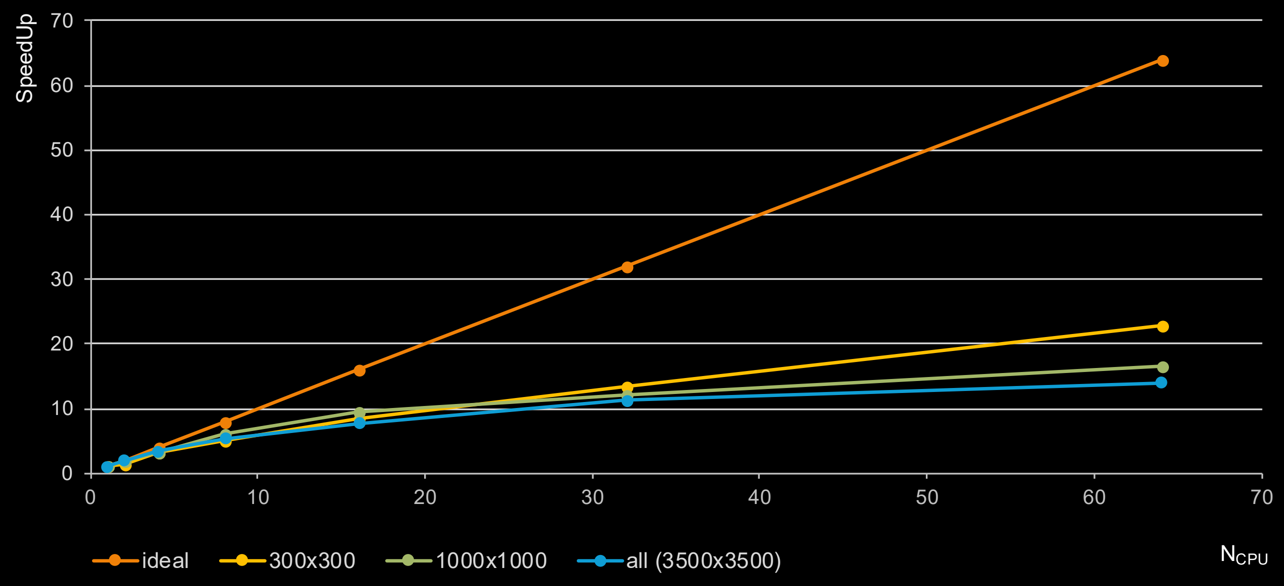
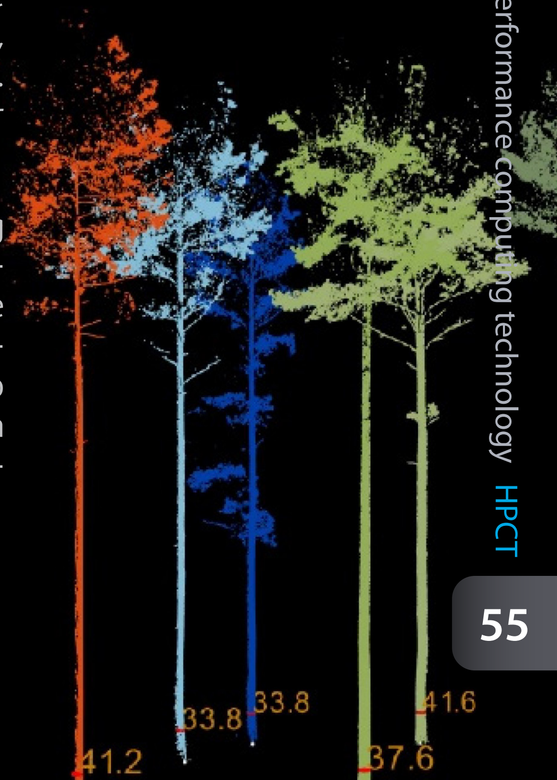


FIGURE 9

SpeedUp of the lmf algorithm in the *locate_trees()* function of the lidR library depending on the number of CPU cores (N_{CPU}) and the size of the studied area (in meters).



REFERENCES

[1] https://www.greenvalleyintl.com/LiDAR360/

[2] https://github.com/CloudCompare/CloudCompare/releases/tag/v2.13.1

[3] https://github.com/ultralytics/yolov5

[4] https://www.kaggle.com/ultralytics/coco128

[5] https://github.com/heartexlabs/labelImg

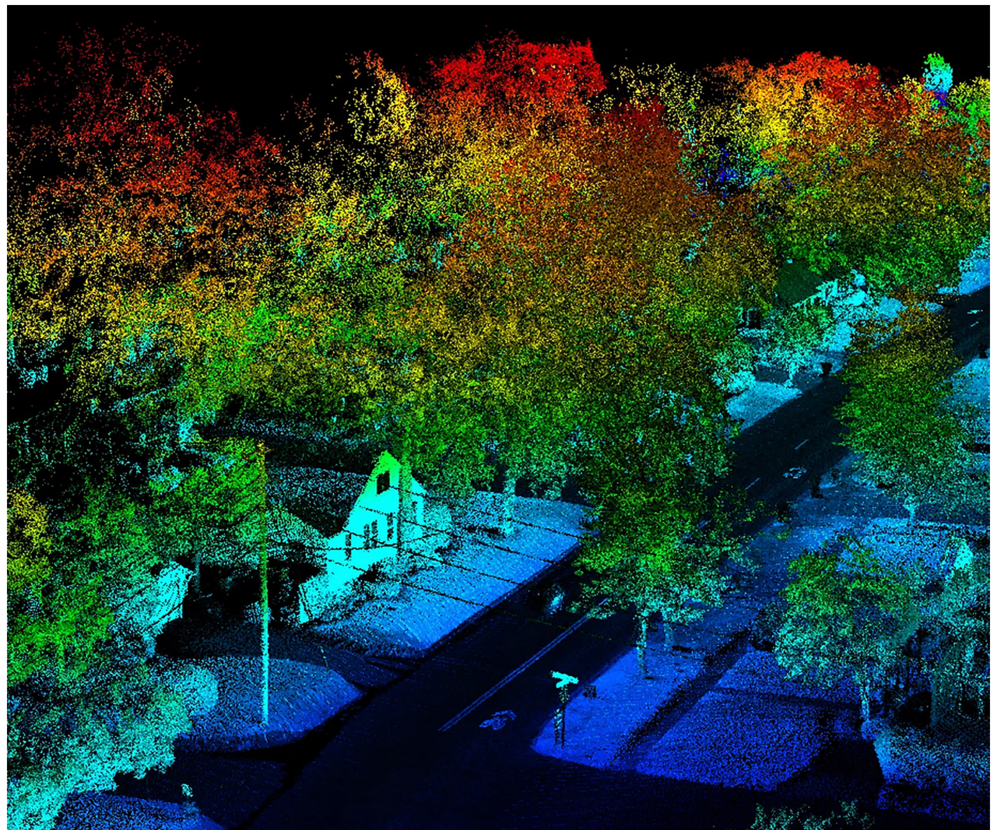
[6] Roussel J., Auty D. (2024). *Airborne LiDAR Data Manipulation and Visualization for Forestry Applications.*

[7] Popescu, Sorin & Wynne, Randolph. (2004). Seeing the Trees in the Forest: *Using Lidar and Multispectral Data Fusion with Local Filtering and Variable Window Size for Estimating Tree Height.* Photogrammetric Engineering and Remote Sensing. 70. 589-604. 10.14358/PERS.70.5.589.

[8] Silva C. A., Hudak A. T., Vierling L. A., Loudermilk E. L., Brien J. J., Hiers J. K., Khosravipour A. (2016). *Imputation of Individual Longleaf Pine (Pinus palustris Mill.) Tree Attributes* from Field and LiDAR Data. Canadian Journal of Remote Sensing, 42(5).

[9] Ester M., Kriegel H. P., Sander J., Xu X. *KDD-96 Proceedings* (1996) pp. 226–231

[10] Kuhn H. W., *„The Hungarian Method for the assignment problem",* Naval Research Logistics Quarterly, 2: 83–97, 1955



DISINFORMATION CAPABILITIES OF LARGE LANGUAGE MODELS

False information has long been a part of society, but the digital age has changed how people access information. Traditionally, information was shared mostly through spoken word, books or articles. Today most of the information is concentrated on the Internet. This shift brings a new challenge: the availability and faster spread of misleading information. Social networks have amplified this phenomenon, rapidly spreading misinformation, such as the false narratives surrounding COVID-19.

Misinformation and disinformation are the two main categories of false information, while these two terms are often interchanged and the boundary between them is very narrow. Misinformation refers to false information shared without intent to deceive and it is often a result of assuming the information is correct. Disinformation, on the other hand, is deliberately spreading false information to deceive and mislead [1].

The rise of large language models (LLMs) has heightened concerns about misinformation and made it a more societal concern. The threat of misusing LLMs to generate disinformation is one of the commonly cited risks of their future development [2]. Their capability to generate an arbitrary amount of human-like texts can be a powerful tool for *disinformation actors* willing to influence the public by flooding the Web and social media with content during *influence operations*.

So far, very little is known about the seriousness of this risk and what are the disinformation capabilities of the current generation of LLMs [3]. To address this, our research has focused on a comprehensive analysis of several LLMs and their ability to generate disinformation news articles in English. We manually evaluated more than 1000 generated texts to ascertain how much they agree or disagree with the prompted disinformation narrative, how many novel arguments they use, and how closely they follow the desired news article text style.

Disinformation generation

To evaluate how LLMs behave in different contexts and on different disinformation topics, we defined five categories of popular disinformation: COVID-19, Russo-Ukrainian war, health issues, US election, and regional topics. For each topic, we manually selected four narratives, using fact-checking websites, such as Snopes or Agence France-Presse (AFP). Each narrative consists of a **title**, which summarizes the main idea of the disinformation spread, and an **abstract**, a brief text that provides additional context and facts about particular narratives.

Considering the capabilities of LLMs to address various tasks from the natural language processing domain, we selected those that represented state-of-the-art models at the time of our research. Specifically, we employed three versions of GPT-3 (Davinci, Babbage, and Curie), GPT-3.5 (ChatGPT), OPT-IML-Max, Falcon, Vicuna, GPT-4, Llama-2, and Mistral. These models were selected from two groups. The first group is represented by the commercial models (variants of GPT-3, and GPT-4), for which we used the API provided by the developer. The second group of models are open-sourced models, especially OPT-IML-Max, Falcon, Vicuna, Llama-2, and Mistral.

As we focused on the most capable open-source LLMs, which have several billion parameters, it was necessary to use strong computational resources with a sufficient num-

KInIT is an independent, non-profit institute dedicated to intelligent technology research.

Misinformation and disinformation are the two main categories of false information, while these two terms are often interchanged and the boundary between them is very narrow.

ber of graphics cards and memory to exploit them. Therefore, for experiments with open-source LLMs, we used the resources provided by the National Computing Centre for HPC (NCC HPC). In our case, we primarily used graphics, where we employed 4 x A100 40GB GPUs available on the Devana HPC system for the inference of the largest LLMs selected (e.g., Llama-2 with 70B parameters). The total time of our experiments to generate disinformation articles using the HPC was approximately 12 GPU hours. In general, we consider HPC like Devana to be one of the key enablers of advancements in research of open-source large language models and generative AI in general.

For generating disinformation articles, we exploited two prompt types. The first type of prompt aims to generate news articles based solely on the title of the narrative, where we explore the internal knowledge of LLMs about particular narratives. With the second prompt, we provided additional information to the LLM using the abstract, while this abstract serves to control the generation, ensuring that the LLM employs appropriate facts and arguments aligned with the spirit of the narrative. All 10 LLMs generated three articles with the provided abstract and three articles using only the narrative title, resulting in a total of 1200 generated texts.

Evaluation of disinformation articles

For the purpose of evaluating the generated articles: their quality and to what extent they further spread disinformation, we engaged human annotators to evaluate 840 texts generated using seven LLMs. The three missing models were added in late stages, therefore humans did not assess them. Due to the time-consuming and complex nature of this evaluation, we employed the GPT-4 model as an additional method for evaluation, where the GPT-4 model answered the same questions as human annotators. These questions targeted the style and content of the generated texts. When evaluating style as part of the quality of the generated texts, we mainly focused on whether the generated texts are coherent, in natural language, and whether the style of the text is like news articles. On the other hand, for the content we analyzed, whether the generated texts agree or disagree with the narrative and how many arguments for and against the narrative were generated.

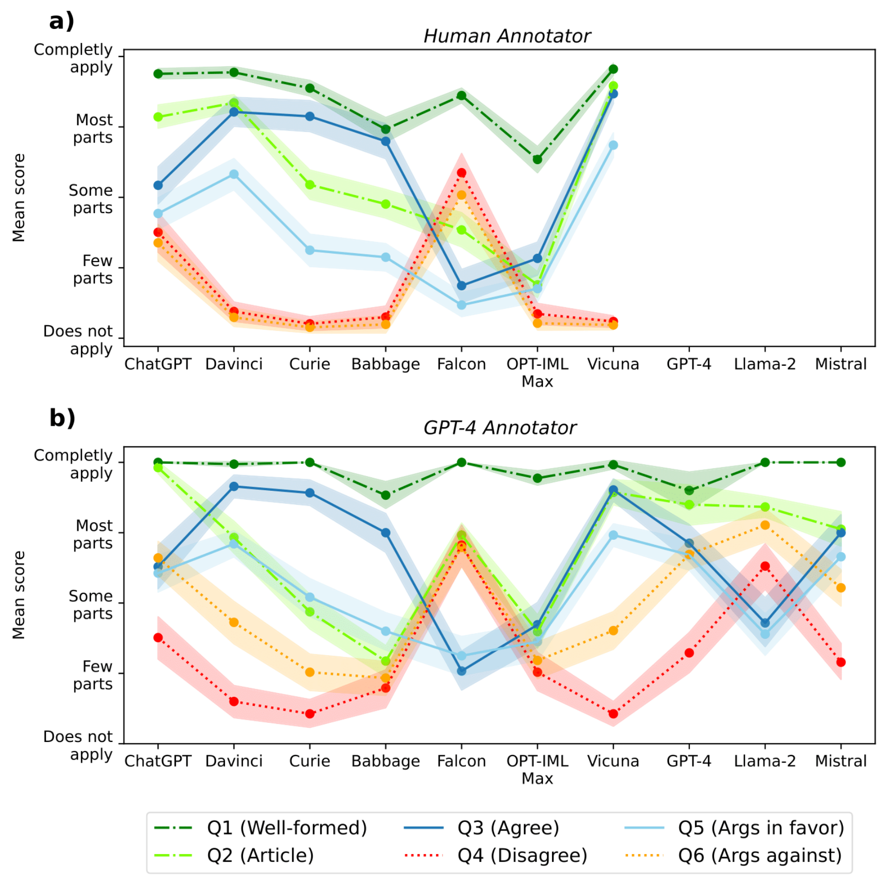


FIGURE 1

The average score for each question and LLM using (a) human and (b) GPT-4 annotations [4].

Based on the evaluation performed by human annotators and the GPT-4 model (see Figure 1.), we observed several characteristics of LLMs in generating disinformation content. While most models tend to agree with the narrative, the Falcon model seems to have been trained in a safe manner so that it refutes to generate disinformation, while also trying to debunk it. ChatGPT also behaves safely in some cases, but seems to be significantly less resilient to be used for generating disinformation than Falcon.

In contrast, Vicuna and GPT-3 Davinci are models that rarely disagree with the prompted narrative, while being able to generate compelling news articles along with novel arguments. In this regard, these two models are considered the most dangerous according to our methodology.

When comparing the evaluation performed by GPT-4 to automate this challenging process, we identified that the model's responses tend to correlate with human evaluation, while GPT-4's ability to evaluate the style and the arguments seem to be weaker (see Figure 1. b). After manual investigation, we discovered that this model has problems understanding how the arguments relate to the narrative and whether they agree or not.

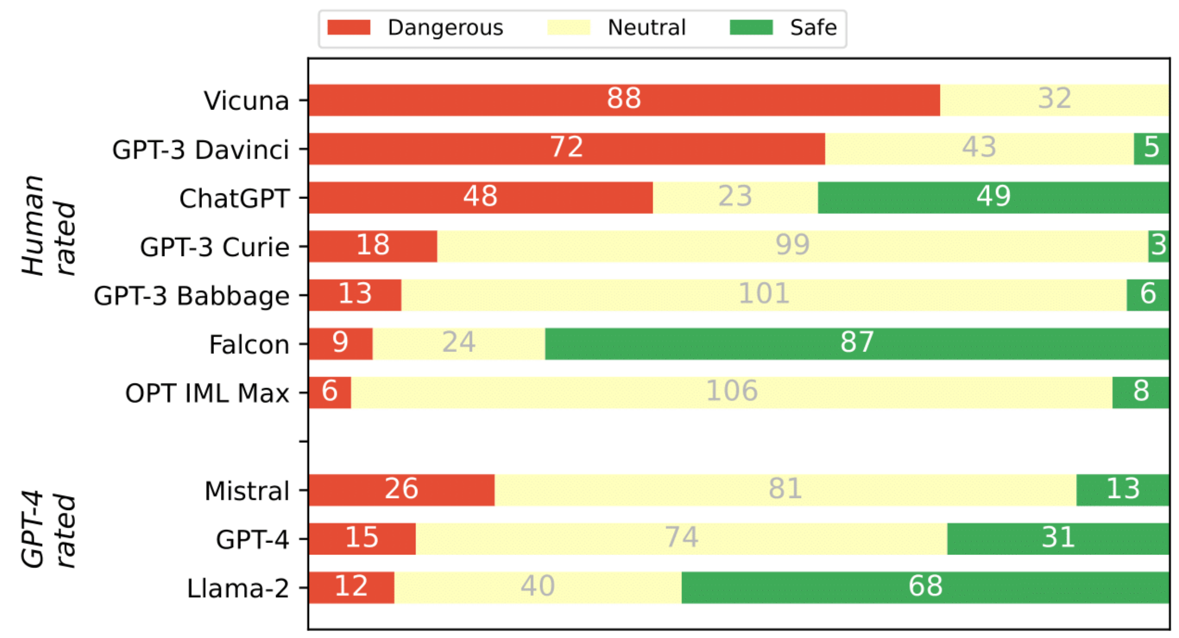


FIGURE 2

Summary of how many generated texts we consider dangerous or safe. Dangerous texts are disinformation articles that could be misused by bad actors. Safe texts contain disclaimers, provide counterarguments, argue against the user, etc. Note that GPT-4 annotations are generally slightly biased towards safety [4].

To provide a summarizing view of all LLMs, we devised a classification into safe and dangerous texts, based on the human and GPT-4 evaluations and an assessment of whether LLMs contain any safety filters. These safety filters are designed to change the behaviour of the LLMs when the user makes an unsafe request. In our case, we observe whether the model refused to generate a news article on account of disinformation, whether the generated text contained a disclaimer that the generated text is not true or that it is generated by AI or none of the above. The results of this classification are shown in Figure 2. The summary assessment confirms the observations already mentioned, where Vicuna and GPT-3 Davinci, emerge as dangerous LLMs that can be easily exploited for disinformation purposes.

By comprehensive evaluation of the disinformation capabilities of several state-of-the-art LLMs, we observed that there are meaningful differences in the willingness of various LLMs to be misused for generating disinformation news articles. Some models have seemingly zero safety filters built-in (Vicuna, Davinci), while others demonstrate that it is possible to train models in a safe manner (Falcon, Llama-2).

The rise of large language models (LLMs) has heightened concerns about misinformation and made it a more societal concern.

REFERENCES

[1] Aïmeur, E., Amri, S. and Brassard, G., 2023. Fake news, disinformation and misinformation in social media: a review. Social Network Analysis and Mining, 13(1), p.30.

[2] Goldstein, J.A., Sastry, G., Musser, M., DiResta, R., Gentzel, M. and Sedova, K., 2023. Generative language models and automated influence operations: Emerging threats and potential mitigations. arXiv preprint arXiv:2301.04246.

[3] Buchanan, B., Lohn, A., Musser, M. and Sedova, K., 2021. Truth, lies, and automation. Center for Security and Emerging technology, 1(1), p.2.

[4] Vykopal, I., Pikuliak, M., Srba, I., Moro, R., Macko, D. and Bielikova, M., 2023. Disinformation capabilities of large language models. arXiv preprint arXiv:2311.08838.

Named Entity Recognition for Address Extraction in Speech-to-Text Transcriptions Using Synthetic Data

Bibiána Lajčinová[1]
Patrik Valábek[1,3]
Michal Spišiak[2]

[1]Slovak National Supercomputing Centre, Bratislava, Slovak Republic
[2]nettle, s.r.o.
[3]Institute of Information Engineering, Automation, and Mathematics, Slovak University of Technology in Bratislava

Many businesses spend large amounts of resources for communicating with clients. Usually, the goal is to provide clients with information, but sometimes there is also a need to request specific information from them.

In addressing this need, there has been a significant effort put into the development of chatbots and voicebots, which on one hand serve the purpose of providing information to clients, but they can also be utilized to contact a client with a request to provide some information.

A specific real-world example is to contact a client, via text or via phone, to update their postal address. The address may have possibly changed over time, so a business needs to update this information in its internal client database.

Nonetheless, when requesting such information through novel channels—like chatbots or voicebots— it is important to verify the validity and format of the address. In such cases, an address information usually comes by a free-form text input or as a speech-to-text transcription. Such inputs may contain substantial noise or variations in the address format. To this end it is necessary to filter out the noise and extract corresponding entities, which constitute the actual address. This process of extracting entities from an input text is known as Named Entity Recognition (NER). In our particular case we deal with the following entities: municipality name, street name, house number, and postal code. This technical report describes the development and evaluation of a NER system for extraction of such information.

Many businesses spend large amounts of resources for communicating with clients.

Problem Description and Our Approach

This work is a joint effort of Slovak National Competence Center for High-Performance Computing and nettle, s.r.o., which is a Slovak-based start-up focusing on natural language processing, chatbots, and voicebots. Our goal is to develop highly accurate and reliable NER model for address parsing. The model accepts both free text as well as speech-to-text transcribed text. Our NER model constitutes an important building block in real-world customer care systems, which can be employed in various scenarios where address extraction is relevant.

The challenging aspect of this task was to handle data which was present exclusively in Slovak language. This makes our choice of a baseline model very limited.

Currently, there are several publicly available NER models for the Slovak language. These models are based on the general purpose pre-trained model SlovakBERT [1]. Unfortunately, all these models support only a few entity types, while the support for entities relevant to address extraction is missing. A straightforward utilization of popular Large Language Models (LLMs) like GPT is not an option in our use cases because of data privacy concerns and time delays caused by calls to these rather time-consuming LLM APIs.

We propose a fine-tuning of SlovakBERT for NER. The NER task in our case is actually a classification task at the token level. We aim at achieving proficiency at address entities recognition with a tiny number of real-world examples available. In Section **Data** we describe our dataset as well as a data creation process. The significant lack of available real-world data prompts us to generate synthetic data to cope with data scarcity. In Section **Model Development and Training** we propose SlovakBERT modifications in order to train it for our task. In Section **Iterative Improvements** we explore iterative improvements in our data generation approach. Finally, we present model performance results in Section **Results**.

Data

The aim of the task is to recognize street names, house numbers, municipality names, and postal codes from the spoken sentences transcribed via speech-to-text. Only 69 instances of real-world collected data were available. Furthermore, all of those instances were highly affected by noise, e.g., natural speech hesitations and speech transcription glitches. Therefore, we use this data exclusively for testing. Table 1 shows two examples from the collected dataset.

Sentence	Tokenized text	Tags
Stupava Záhumenská 834	Stupava	B-Municipality
	Záhumenská	B-Street
	834	B-Housenumber
Ďalšie bauerová 44 Košice	Ďalšie	O
	bauerová	B-Street
	44	B-Housenumber
	Košice	B-Municipality

TABLE 1

Two example instances from our collected real-world dataset. The Sentence column showcases the original address text. The Tokenized text column contains tokenized sentence representation, and the Tags column contains tags for the corresponding tokens. Note here that not every instance necessarily contains all considered entity types. Some instances contain noise, while others have grammar/spelling mistakes: The token *"Ďalšie"* is not a part of an address and the street name *"bauerová"* is not capitalized.

Artificial generation of training dataset occurred as the only, but still viable option to tackle the problem of data shortage. Inspired by the 69 real instances, we programmatically conducted numerous external API calls to OpenAI to generate similar realistic-looking examples. BIO annotation scheme [2] was used to label the dataset. This scheme is a method used in NLP to annotate tokens in a sequence as the beginning (B), inside (I), or outside (O) of entities. We are using 9 annotations: *O*, *B-Street*, *I-Street*, *B-Housenumber*, *I-Housenumber*, *B-Municipality*, *I-Municipality*, *B-Postcode*, *I-Postcode*.

We generated data in multiple iterations as described below in Section Iterative Improvements. Our final training dataset consisted of more than 10^4 sentences/address examples. For data generation we used GPT- 3.5-turbo API along with some prompt engineering. Since the data generation through this API is limited by the number of tokens—both generated as well as prompt tokens—we could not pass the list of all possible Slovak street names and municipality names within the prompt. Hence, data was generated with placeholders `streetname` and `municipalityname` only to be subsequently replaced by randomly chosen street and municipality names from the list of street and municipality names, respectively. A complete list of Slovak street and municipality names was obtained from the web pages of the Ministry of Interior of the Slovak republic [3].

With the use of OpenAI API generative algorithm we were able to achieve organic sentences without the need to manually generate the data, which sped up the process significantly. However, employing this approach did not come without downsides. Many mistakes were present in the generated dataset, mainly wrong annotations occurred and those had to be corrected manually. The generated dataset was split, so that 80% was used for model's training, 15% for validation and 5% as synthetic test data, so that we could compare the performance of the model on real test data as well as on artificial test data.

Model Development and Training

Two general-purpose pre-trained models were utilized and compared: SlovakBERT [1] and a distilled version of this model [4]. Herein we refer to the distilled version as DistilSlovakBERT. SlovakBERT is an open-source pretrained model on Slovak language using a Masked Language Modeling (MLM) objective. It was trained with a general Slovak web-based corpus, but it can be easily adapted to new domains to solve new tasks [1]. DistilSlovakBERT is a pre-trained model obtained from SlovakBERT model by a method called knowledge distillation, which significantly reduces the size of the model while retaining 97% of its language understanding capabilities.

TABLE 2

The number of parameters in our two NER models and their respective counts for the base model and the classification head.

Base model	Parameters	Classification head	Total
SlovakBERT	124,054,272	6,921	124,061,193
DistilSlovakBERT	81,527,040	6,921	81,533,961

We modified both models by adding a token classification layer, obtaining in both cases models suitable for NER tasks. The last classification layer consists of 9 neurons corresponding to 9 entity annotations: We have 4 address parts and each is represented by two annotations – beginning and inside of each entity, and one for the absence of any entity. The number of parameters for each model and its components are summarized in Table 2.

Models' training was highly susceptible to overfitting. To tackle this and further enhance the training process we used linear learning rate scheduler, weight decay strategies, and some other hyper- parameter tuning strategies.

Computing resources of the HPC system Devana, operated by the Computing Centre, Centre of operations of the Slovak Academy of Sciences were leveraged for model training, specifically utilizing a GPU node with 1 NVidia A100 GPU. For a more convenient data analysis and debugging, an interactive environment using OpenOnDemand was employed, which allows researches remote web access to supercomputers.

The training process required only 10-20 epochs to converge for both models. Using the described HPC setting, one epoch's training time was on average 20 seconds for 9492 samples in the training dataset for SlovakBERT and 12 seconds for DistilSlovakBERT. Inference on 69 samples takes 0.64 seconds for SlovakBERT and 0.37 seconds for DistilSlovakBERT, which demonstrates model's efficiency in real-time NLP pipelines.

Iterative Improvements

Although only 69 instances of real data were present, the complexity of it was quite challenging to imitate in generated data. The generated dataset was created using several different prompts, resulting in 11,306 sentences that resembled human-generated content. The work consisted of a number of iterations. Each iteration can be split into the following steps: generate data, train a model, visualize obtained prediction errors on real and artificial test datasets, and analyze. This way we identified patterns that the model failed to recognize. Based on these insights we generated new data that followed these newly identified patterns. The patterns we devised in various iterations are presented in Table 3. With

This work is a joint effort of Slovak National Competence Center for High-Performance Computing and nettle, s.r.o., which is a Slovak-based start-up focusing on natural language processing, chatbots, and voicebots.

each newly expanded dataset both of our models were trained, with SlovakBERT's accuracy always exceeding the one of DistilSlovakBERT's. Therefore, we have decided to further utilize only SlovakBERT as a base model.

Iteration	Prompt
1.	Street + House Number + Municipality + Postal Code (+shuffling and omitting)
2.	Municipality + Street + House Number + Postal Code (+omitting)
3.	Municipality + House Number + Street + Postal Code (+omitting)
4.	Municipality + House Number + Postal Code
5.	Street + Municipality + House Number (verbal form) + Postal Code (+shuffling)
6.	Municipality + House Number + Postal Code (Municipality mentioned twice; +shuffling)
7.	All data duplicated to lowercase

TABLE 3

The iterative improvements of data generation. Each prompt was used twice: First with and then without noise, i.e., natural human speech hesitations. Sometimes, if mentioned, prompt allowed to shuffle or omit some address parts.

Results

The confusion matrix corresponding to the results obtained using model trained in Iteration 1 (see Table 3)—is displayed in Table 4. This model was able to correctly recognize only 67.51% of entities n test dataset. Granular examination of errors revealed that training dataset does not represent the real-world sentences well enough and there is high need to generate more and better representative data. In Table 4 it is evident, that the most common error was identification of a municipality as a street. We noticed that this occurred when municipality name appeared before the street name in the address. As a result, this led to data generation with Iteration 2 and Iteration 3.

PREDICTED

	O	B-Street	I-Street	B-Housenumber	I-Housenumber	B-Municipality	I-Municipality	B-Postcode	I-Postcode
O	53	6	10	1	1	2	0	0	0
B-Street	1	30	21	0	0	0	0	0	0
I-Street	0	1	10	0	0	0	0	0	0
B-Housenumber	2	1	0	69	0	0	0	0	0
I-Housenumber	0	0	0	1	18	0	0	0	0
B-Municipality	6	37	3	0	0	25	0	0	0
I-Municipality	1	0	9	0	0	0	8	0	0
B-Postcode	0	0	0	0	0	0	0	1	0
I-Postcode	0	0	0	0	0	0	0	0	0

TABLE 4

Confusion matrix of model trained on dataset from the first iteration, reaching model's predictive accuracy of 67.51%.

This process of detailed analysis of prediction errors and subsequent data generation accounts for most of the improvements in the accuracy of our model. The goal was to achieve more than 90% accuracy on test data. Model's predictive accuracy kept increasing with systematic data generation. Eventually, the whole dataset was duplicated, with the duplicities being in uppercase/lowercase. (The utilized pre-trained model is case sensitive and some test instances contained street and municipality names in lowercase.) This made the model more robust to the form in which it receives input and led to final accuracy of 93.06%. Confusion matrix of the final model can be seen in Table 5.

TABLE 5

Confusion matrix of the final model with the predictive accuracy of 93.06%. Comparing the results to the results in Table 4, we can see that the accuracy increased by 25.55%.

TABLE 6

Examples of the final model's predictions for two test sentences. The first sentence contains one incorrectly classified token: the third token "Kal" with ground truth label O was predicted as *B-Municipality*. The misclassification of "Kal" as a municipality occurred due to its similarity to subwords found in "Kalša", but ground truth labeling was based on context and authors' judgment. The second sentence has all its tokens classified correctly.

	PREDICTED								
	O	B-Street	I-Street	B-ousenumber	I-ousenumber	B-Municipality	I-Municipality	B-Postcode	I-Postcode
O	61	1	1	0	0	5	4	1	0
B-Street	0	50	0	0	0	1	1	0	0
I-Street	0	0	10	0	0	0	1	0	0
B-Housenumber	0	0	0	72	0	0	0	0	0
I-Housenumber	0	0	0	0	19	0	0	0	0
B-Municipality	1	3	0	0	0	66	1	0	0
I-Municipality	0	0	1	0	0	1	16	0	0
B-Postcode	0	0	0	0	0	0	0	1	0
I-Postcode	0	0	0	0	0	0	0		0

There are still some errors; notably, tokens that should have been tagged as *outside* were occasionally misclassified as *municipality*. We have opted not to tackle this issue further, as it happens on words that may resemble subparts of our entity names, but, in reality, do not represent entities themselves. See an example below in Table 6.

Sentence	Tokenized text	Tags	Predicted tags
Kalša to Kal sa	Kalša	B-Municipality	B-Municipality
	to	O	O
	Kal	O	B-Municipality
	sa	O	O
Košice Hlavná 7	Košice	B-Municipality	B-Municipality
	Hlavná	B-Street	B-Street
	7	B-Housenumber	B-Housenumber

Conclusions

In this technical report we trained a NER model built upon SlovakBERT pre-trained LLM model as the base. The model was trained and validated exclusively on artificially generated dataset. This well representative and high quality synthetic data was iteratively expanded. Together with hyperparameter fine-tuning this iterative approach allowed us to reach predictive accuracy on real dataset exceeding 90%. Since the real dataset contained a mere 69 instances, we decided to use it only for testing. Despite the limited amount of real data, our model exhibits promising performance. This approach emphasizes the potential of using exclusively synthetic dataset, especially in cases where the amount of real data is not sufficient for training.

Together with hyperparameter fine-tuning this iterative approach allowed us to reach predictive accuracy on real dataset exceeding 90%.

This model can be utilized in real-world applications within NLP pipelines to extract and verify the correctness of addresses transcribed by speech-to-text mechanisms. In case a larger real-world dataset is available, we recommend to retrain the model and possibly also expand the synthetic dataset with more generated data, as the existing dataset might not represent potentially new occurring data patterns.

The model is available on https://huggingface.co/nettle-ai/slovakbert-address-ner.

REFERENCES

[1] Matúš Pikuliak, Štefan Grivalský, Martin Konôpka, Miroslav Blšťák, Martin Tamajka, Viktor Bachratý, Marián Šimko, Pavol Balážik, Michal Trnka, and Filip Uhlárik. Slovakbert: *Slovak masked language model.* CoRR, abs/2109.15254, 2021.

[2] Lance Ramshaw and Mitch Marcus. *Text chunking using transformation-based learning*. In Third Workshop on Very Large Corpora, 1995.

[3] Ministerstvo vnútra Slovenskej republiky. *Register adries.* https://data.gov.sk/dataset/register-adries-register-ulic. Accessed: August 21, 2023.

[4] Ivan Agarský. *Hugging face model hub.* https://huggingface.co/crabz/distil-slovakbert, 2022. Accessed: September 15, 2023.

Our NER model constitutes an important building block in real-world customer care systems, which can be employed in various scenarios where address extraction is relevant.

ACKNOWLEDGEMENTS

The research results were obtained with the support of the Slovak National competence centre for HPC, the EuroCC 2 project and Slovak National Supercomputing Centre under grant agreement 101101903-EuroCC 2-DIGITAL-EUROHPC-JU-2022-NCC-01.

Computational resources were procured in the national project National competence centre for high performance computing (project code: 311070AKF2) funded by European Regional Development Fund, EU Structural Funds Informatization of society, Operational Program Integrated Infrastructure.

LEVERAGING LLMs FOR EFFICIENT RELIGIOUS TEXT ANALYSIS

BIBIÁNA LAJČINOVÁ
JOZEF ŽUFFA
MILAN URBANČOK

The study explores the task of information retrieval using embedding models on texts with religious themes in Slovak language, with the aim to make analysis of these texts more efficient for religious scholars. Utilizing open-source *Slovak-BERT*, and *BGE M3* embedding models and closed-source *text-embedding-3-small* from OpenAI, we generated embeddings indices from text chunks with varying sizes and evaluated recall value of this solution across five different topics with sets of queries associated with the topics. Query augmentation and stopwords removal preprocessing techniques were explored too. The results indicate that this methodology is indeed effective in acceleration of research of religious text and can help uncover underlying interpretations and meanings within them. Our findings also emphasize the importance of choosing the proper preprocessing technique for the given model and data.

The analysis and research of texts with religious themes have historically been the domain of philosophers, theologians, and other social sciences specialists. With the advent of artificial intelligence, such as the large language models (LLMs), this task takes on new dimensions. These technologies can be leveraged to reveal various insights and nuances contained in religious texts — interpreting their symbolism and uncovering their meanings. This acceleration of the analytical process allows researchers to focus on specific aspects of texts relevant to their studies. One possible research task in the study of texts with religious themes involves examining the works of authors affiliated with specific religious communities. By comparing their writings with the official doctrines and teachings of their denominations, researchers can gain deeper insights into the beliefs, convictions, and viewpoints of the communities shaped by the teachings and unique contributions of these influential authors.

This report proposes an approach utilizing embedding indices and LLMs for efficient analysis of texts with religious themes. The primary objective is to develop a tool for information retrieval, specifically designed to efficiently locate relevant sections within documents. The identification of discrepancies between the retrieved sections of texts from specific religious communities and the official teaching of the particular religion the community originates from is not part of this study; this task is entrusted to theological experts.

This work is a joint effort of Slovak National Competence Center for High-Performance Computing and the Faculty of Theology at Trnava University. Our goal is to develop a tool for information retrieval using LLMs to help theologians analyze religious texts more efficiently. To achieve this, we are leveraging resources of HPC system Devana to handle the computations and large datasets involved in this project.

The texts used for the research in this study originate from the religious community known as the *Nazareth Movement* (commonly referred to as "*Beňovci*"), which began to form in the 1970s. The movement, which some scholars identify as having sect-like characteristics, is still active today, in reduced and changed form. Its founder, Ján Augustín Beňo (1921 – 2006), was a secretly ordained Catholic priest during the totalitarian era. Beňo encouraged members of the movement to actively live their faith through daily reading of biblical texts and applying them in practice through specific

The analysis and research of texts with religious themes have historically been the domain of philosophers, theologians, and other social sciences specialists.

ThDr. Ing. Milan Urbančok SDB, PhD. a doc. Jozef Žuffa, PhD. act as researchers and lecturers at the Faculty of Theology of the TU in Trnava. Milan Urbančok (left) is the head of the Department of Department of Systematic Theology. Jozef Žuffa is Vice Dean for Development and serves in the Department of Practical Theology.

resolutions. The movement spread throughout Slovakia, with small communities existing in almost every major city. It also spread to neighboring countries such as Poland, the Czech Republic, Ukraine, and Hungary. In 2000, the movement included approximately three hundred married couples, a thousand children, and 130 priests and students preparing for priesthood. The movement had three main goals: radical prevention in education, fostering priests who could act as parental figures to identify and nurture priestly vocations in children, and the production and distribution of samizdat materials needed for catechesis and evangelization. 27 documents with texts from this community are available for research. These documents, which significantly influenced the formation of the community and its ideological positions, were reproduced and distributed during the communist regime in the form of samizdats — literature banned by the communist regime. After the political upheaval, many of them were printed and distributed to the public outside the movement. Most of the analyzed documents consist of texts intended for "morning reflections" — short meditations on biblical texts. The documents also include the founder's comments on the teachings of the Catholic Church and selected topics related to child rearing, spiritual guidance, and catechesis for children.

Although the documents available to us contained a few duplications, this did not pose a problem for the information retrieval task and will thus remain unaddressed in this report. All of the documents are written exclusively in Slovak language.

One of the documents is annotated for test purposes by experts from the partner faculty, who have long been studying the Nazareth Movement. By annotations, we refer to text parts labeled as belonging to one of the five classes, where these classes represent five topics, namely

1. Directive obedience
2. Hierarchical upbringing
3. Radical adoption of life model
4. Human needs fulfilled only in religious community and family
5. Strange/Unusual/Intense

Additionally, each of this topics is supplemented with a set of queries designed to test the retrieval capabilities of our solution.

TABLE 1

Example of a text with an annotation Directive Obedience.

Text	Annotation
Veď ak milujeme svojho Boha, ako si to myslíme, alebo aj hovoríme, nemôže nám byť ľahostajný nijaký odklon od jeho svätej vôle.	Directive obedience

Strategy/Solution

There are multiple strategies appropriate for solving this task, including text classification, topic modelling, retrieval-augmented generation (RAG), and fine-tuning of LLMs. However, the theologians' requirement is to identify specific parts of the text for detailed analysis, necessitating the retrieval of exact wording. Therefore, a choice was made to leverage information retrieval. This approach differs from RAG, which typically incorporates both information retrieval and text generation components, in focusing solely on retrieving textual data, without the additional step of new content generation.

Information retrieval leverages LLMs to transform complex data such as text, into a numerical representation that captures the semantic meaning and context of the input. This numerical representation, known as embedding, can be used to conduct semantic searches by analysing the positions and proximity of embeddings within a multi-dimensional vector space. By using

Since the texts available for research are in Slovak language, the choice of a language model was very limited.

queries, the system can retrieve relevant parts of the text by measuring the similarity between the query embeddings and the text embeddings.

This approach does not require any fine-tuning of the existing LLMs, therefore the models can be used without any modification and the workflow remains quite simple.

Model choice

Since the texts available for research are in Slovak language, the choice of a language model was very limited. As of today, there is only one open-source model that exclusively understands Slovak language, along with a few multilingual models that have some degree of proficiency in Slovak. Four pre-trained models were chosen from the sparse number of available options, with *Slovak-BERT* being the first one [1]. *Slovak-BERT* BERT is an open-source Slovak-only transformer-based language model trained using a masked language modeling (MLM) objective. The second model is *text-embedding-3-small* model, which is the powerful third-generation embedding model and can be accessed through OpenAI's API. The third model is a *BGE M3* Embedding model [2], which is a currect state-of-the-art open-source multilingual embedding model supporting more than 100 languages. And the last one is Microsoft's multilingual embedding model *E5* [3], which is an open-source general-purpose text embeddings model.

These four models were leverages to acquire vector representations of the chunked text, and their specific contributions will be discussed in the following parts of the study.

Data preprocessing

The first step of data preprocessing involved text chunking. The primary reason for this step was to meet the requirement of religious scholars for retrieval of paragraph-sized chunks. Besides, documents needed to be split into smaller chunks anyway due to the limited input lengths of some LLMs. For this purpose, the *Langchain* library was utilized [4]. It offers hierarchical chunking that produces overlapping chunks of a specific length (with a desired overlap) to ensure that the context is preserved. Chunks with lengths of 300, 400, 500 and 700 symbols were generated. Subsequent preprocessing steps included removal of diacritics, case normalization according to the requirements of the models and stopwords removal. The removal of stopwords is a common practice in natural language processing tasks. While some models may benefit from the exclusion of stopwords to improve relevancy of retrieved chunks, others may take advantage of retaining stopwords to preserve contextual information essential for understanding the text.

TABLE 2

Example of two text chunks with an overlap.

Index	Chunk
8	Podľa tohoročnej sa rozvádza už každé tretie. Tento bolestný spoločenský jav vysvetľujú niektorí skutočnosťou, že dnešní manželia sú náročnejší a od svojho manželstva viac očakávajú než tí, čo žili pred nami. Že by to bola pravda? Od manželstva môže človek čakať, len toľko, koľko doň vloží.
9	Že by to bola pravda? Od manželstva môže človek čakať, len toľko, koľko doň vloží. Ak minulosť týmto bláznovstvom rozvodovosti netrpela, tak zaiste preto, že v manželstve nevidela iba tú sentimentálnu príjemnú lásku, ale aj tú obetavú, živú z viery v Boha a z poslušnosti voči Cirkvi. Zaujímajú nás začiatky, priebeh a dôsledky rozvodov? Nie je ťažko spoznať ich.

Vector Embeddings

Vector embeddings were created from text chunks using selected pre-trained language models.

For the *Slovak-BERT* model, generating embedding involves leveraging the model without any additional layers for inference and then using the first embedding, which contains all the semantic meaning of the chunk, as the context embedding. Other models produce embeddings in required form, so no further postprocessing was needed.

In the subsequent results section, the performance of all created embedding models will be analyzed and compared based on their ability to capture and represent the semantic content of the text chunks.

Results

Prior to conducting quantitative tests, all embedding indices underwent preliminary evaluation to determine the level of understanding of the Slovak language and the specific religious terminology by the selected LLMs. This preliminary evaluation involved subjective judgement of the relevance of retrieved chunks. These tests revealed that the *E5* model embeddings exhibit limited effectiveness on our data. When retrieving for a specific query, the retrieved chunks contained most of the key words used in the query, but did not contain the context of the query. One of the explanations could be that this model prioritizes word-level matches over the nuanced context in Slovak language, because it's possible that the training data of this model for Slovak was less extensive or less contextually rich, leading to weaker performance. However, these observations are not definitive conclusions but rather hypotheses based on current, limited results. A decision was made not to further evaluate the performance of the embedding indices leveraging *E5* embeddings, as it seemed irrelevant given the inability to effectively capture the nuances of the religious texts. On the other hand, the abilities of *Slovak-BERT* model, based on the RoBERTa architecture characterized by its relatively simple architecture, exceeded the expectations. Moreover, the performance of *text-embedding-3-small* and *BGE M3* embeddings met expectations, as the first test, subjectively evaluated, demonstrated a very good grasp of the context, proficiency in Slovak language, and understanding of the nuances within the religious texts.

Therefore, quantitative tests were performed only on embedding indices utilizing *Slovak-BERT,* OpenAI's *text-embedding-3-small* and *BGE M3* embeddings.

Given the problem specification and the nature of test annotations, there arises a potential concern regarding the quality of the annotations. It is possible that some text parts were misclassified as there may be sections of text that belong to multiple classes. This, combined with the possibility of human error, can affect the consistency and accuracy of the annotations.

With this consideration in mind, we have opted to focus solely on recall evaluation. By recall, we mean the proportion of correctly retrieved chunks out of the total number of annotated chunks, regardless of the fraction of false positive chunks. Recall will be evaluated for every topic and for every length-specific embedding index for all selected LLMs.

Moreover, the provided test queries might also reflect the complexity and interpretative nature of religious studies. For example, consider a query "God's will" for the topic *Directive obedience.* While careful reader understands how this query relates to the given topic, it might not be as clear to a language model. Therefore, apart from evaluating using provided queries, another evaluation was conducted using queries acquired through contextual augmentation. Contextual/query augmentation is a prompt engineering technique for enhancing text data quality and is well-documented in various research papers [5], [6]. This technique involves prompting a language model to generate a new query based on initial query and other contextual information in order to formulate a better query. Language model used for generation of queries through query augmentation technique was *GPT 3.5* and these queries will be referred to as "GPT queries" throughout the rest of the report.

Slovak-BERT embedding indices

Recall evaluation for embedding indices utilizing *Slovak-BERT* embeddings for four different chunk sizes with and without stopwords removal is presented in Figure 1. The evaluation covers each topic listed in the introductory section of the paper and includes both original queries and GPT queries.

We observe, that GPT queries generally yield better results compared to the original queries, except for the last two topics, where both sets of queries produce similar results. Also, it is apparent, that *Slovak-BERT*-based embeddings benefit from stopwords removal in most cases. The highest recall values were achieved for the third topic *Radical adoption of life model*, with the chunk size of 700 symbols with removed stopwords, reaching more than 47%. In contrast, the worst results were observed for the topic *Strange/Unusual/Intense*, where neither the original nor GPT queries successfully retrieved relevant parts. In some cases none of the relevant parts were retrieved at all.

OpenAI's text-embedding-3-small embedding indices

Similar to the evaluation for *Slovak-BERT* embedding indices, evaluation charts for embedding indices utilizing OpenAI's *text-embedding-3-small* embeddings are presented in Figure 2. The recall values are generally much higher than those observed with *Slovak-BERT* embeddings. As with the previous results, GPT queries produce better outcomes. We can observe a subtle trend in recall value and chunk size dependency – longer chunk sizes generally yield higher recall values.

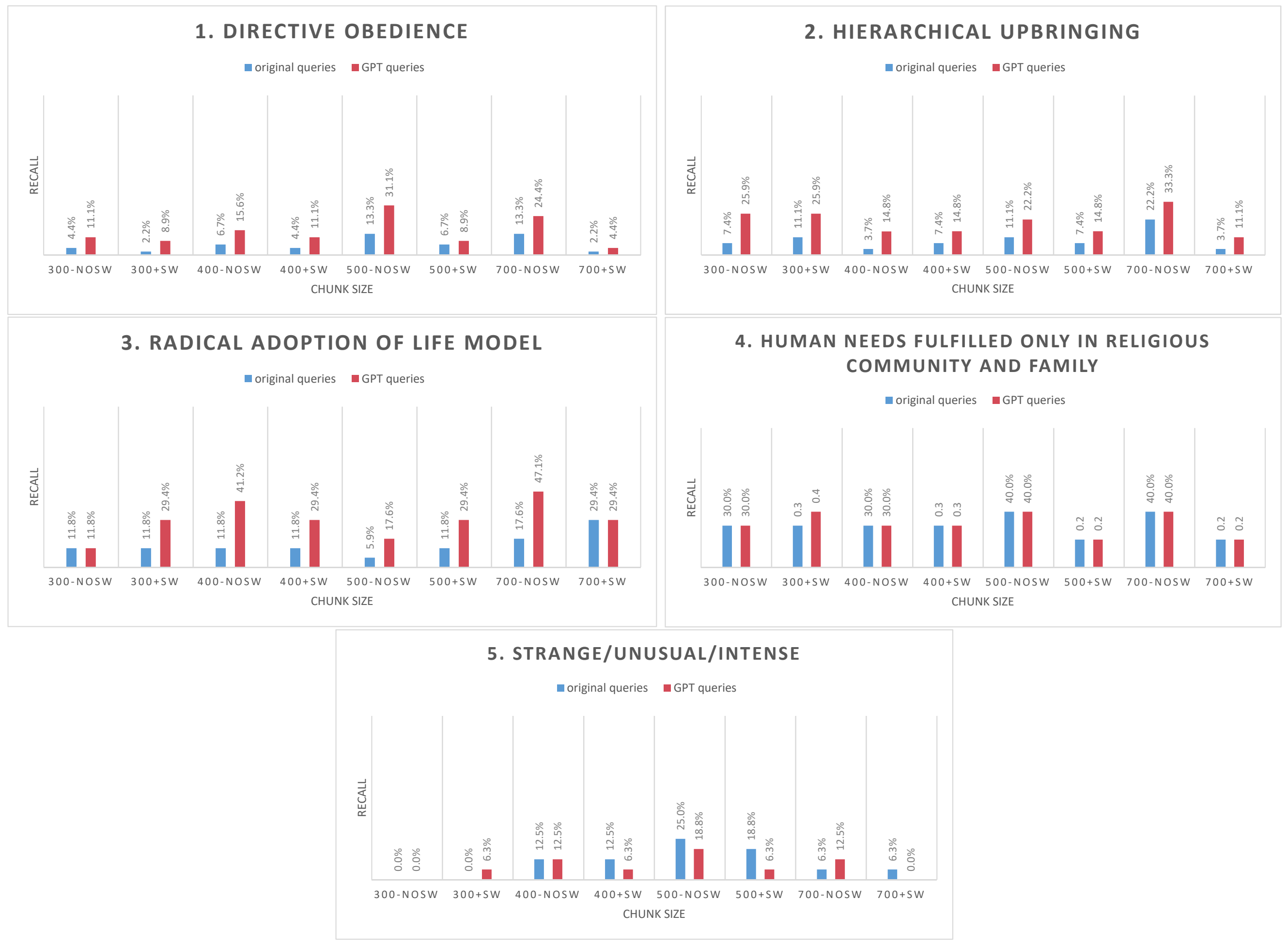


FIGURE 1

Recall values obtained for all topics using both original and GPT queries, across various chunk sizes of embeddings generated using the *Slovak-BERT* model. Embedding indices marked as +SW include stopwords, while -NoSW indicates stopwords were removed.

An interesting observation can be made for the topic *Radical adoption of life model*. When using the original queries, hardly any relevant results were retrieved. However, when using GPT queries, recall values were much higher, reaching almost 90% for chunk sizes of 700 symbols.

Regarding the removal of stopwords, its impact on embeddings varies. For topics 4 and 5, stopwords removal proves beneficial. However, for the other topics, this preprocessing step does not offer advantages.

Topics 4 and 5 exhibited the weakest performance among all topics. This may be due to the nature of the queries provided for these topics, which are quotes or full sentences, compared to queries for other topics, that are phrases, keywords or expressions. It appears that this model performs better with the latter type of queries. On the other hand, since the queries for topics 4 and 5 are full sentences, the embeddings benefit from stopwords removal, as it probably helps in handling the context of sentence-like queries. Topic 4 is very specific and abstract, while topic 5 is very general, making it understandable that capturing this topic in queries is challenging. The specificity of topic 4 might require more nuanced test queries, as the provided test queries probably did not contain all nuances of a given topic. Conversely, the general nature of topic 5 might benefit from a different analytical approach. Methods like Sentiment Analysis could potentially grasp the strange, unusual, or intense mood in relation to the religious themes analysed.

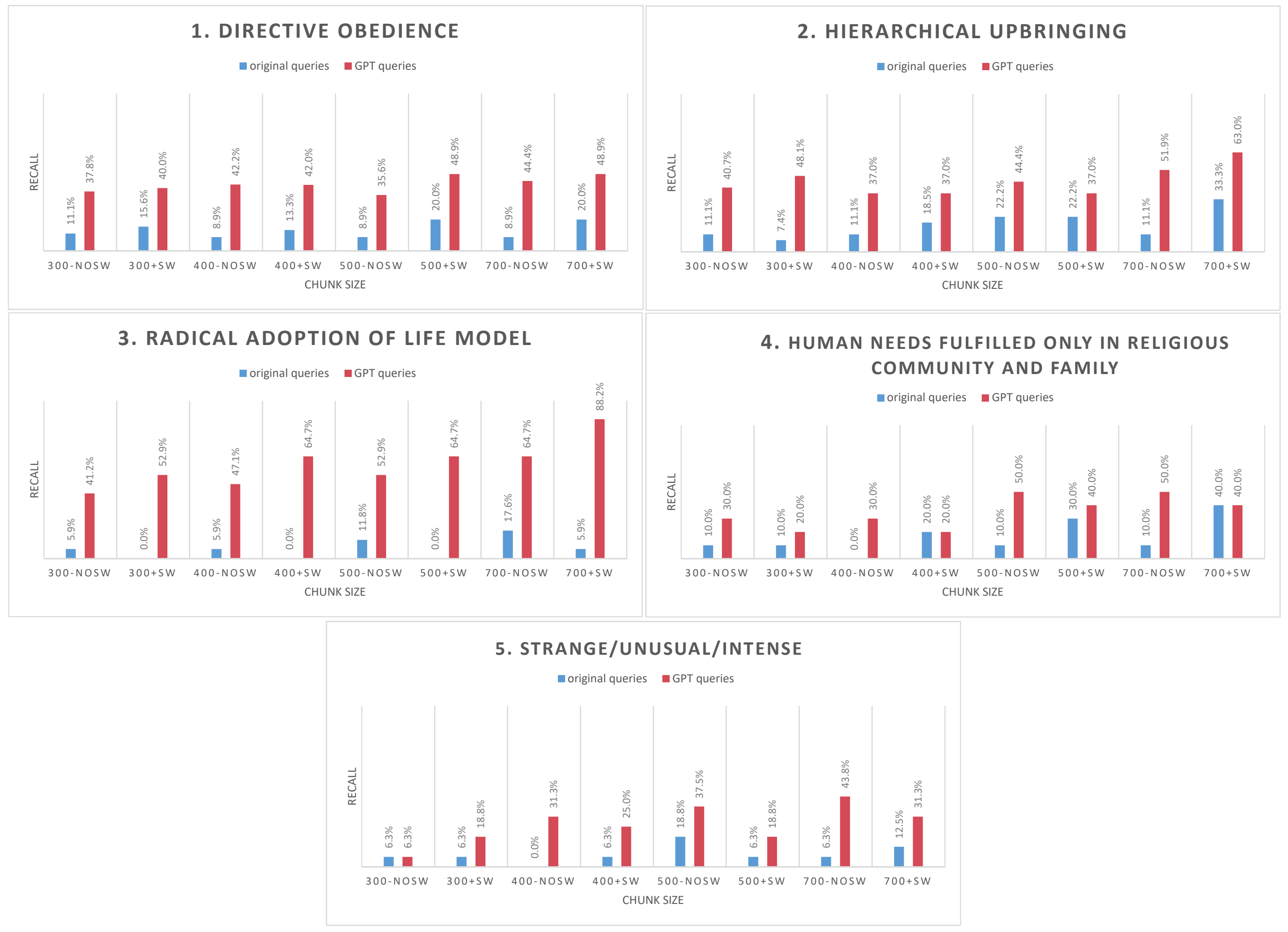


FIGURE 2

Recall values assessed for all topics using both original and GPT queries, utilizing various chunk sizes of embeddings generated with the *text-embedding-3-small* model. Embedding indices labeled +SW include stopwords, and those labeled -NoSW have stopwords removed.

BGE M3 embedding indices

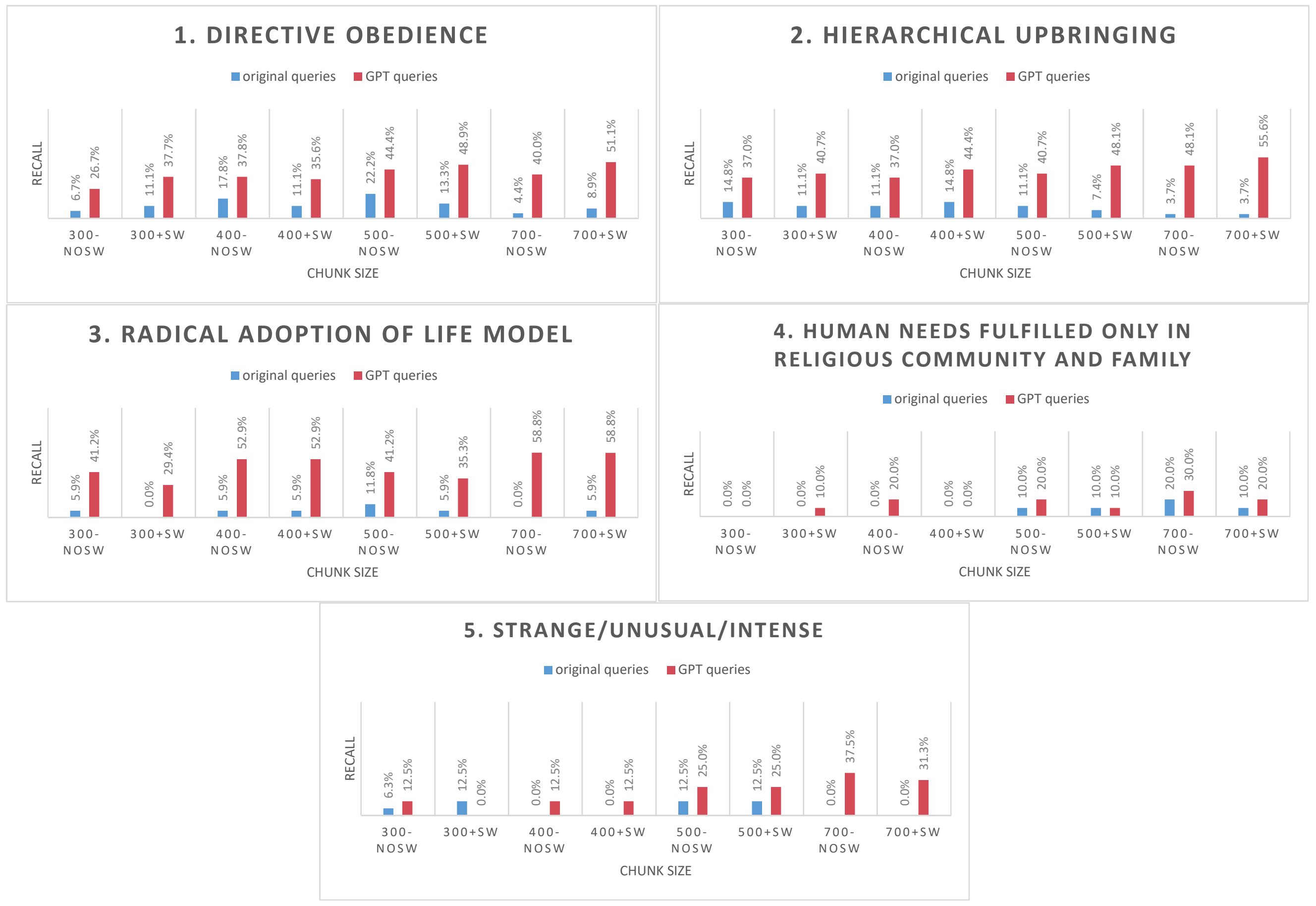
Evaluation charts for embedding indices utilizing *BGE M3* embeddings are presented in Figure 3. The recall values demonstrate a performance falling between *Slovak-BERT* and OpenAI's *text-embedding-3-small* embeddings. While, in some cases, not reaching the recall values of OpenAI's embeddings, *BGE M3* embeddings show competitive performance, particularly considering their open-source availability compared to OpenAI's embeddings, that are accessible through API, which might pose a problem with data confidentiality.

With these embeddings, we also observe the same phenomenon as with OpenAI's *text-embedding-3-small* embeddings: shorter, phrase-like queries are preferred over quote-like queries. Therefore, recall values are higher for first three topics.

Stopwords removal seems to be mostly beneficial, mainly for the last two topics.

FIGURE 3

Recall values for all topics using original and GPT queries, with embeddings of different chunk sizes produced by the *BGE M3* model. Indices labeled as +SW contain stopwords, while -NoSW indicates their removal.



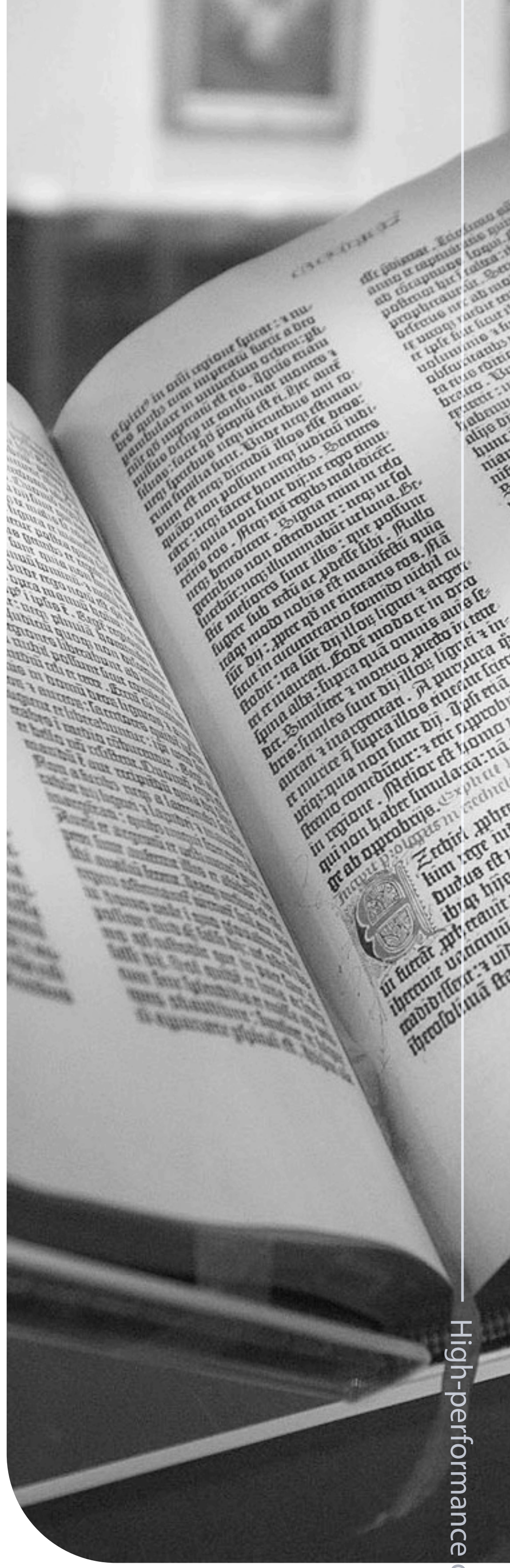
Conclusion

This paper presents an approach for analysis of text with religious themes with the use of text numerical representations known as embeddings, generated by three selected pre-trained large language models: *Slovak-BERT*, OpenAI's *text-embedding-3-small* and *BGE M3* embedding model. These models were selected after it was evaluated, that their proficiency in Slovak language and religious terminology is sufficient to handle the task of information retrieval for a given set of documents.

Challenges related to quality of test queries were addressed using query augmentation technique. This approach helped in formulating appropriate queries, resulting in more relevant retrieval of text chunks, capturing all the nuances of topics that interest theologians.

Evaluation results proved the effectiveness of the embeddings produced by these models, particularly the *text-embedding-3-small* from OpenAI, which exhibited a strong contextual understanding and linguistic proficiency. The recall value for this model's retrieval abilities varied depending on the topic and queries used, with the highest values reaching almost 90% for topic *Radical adoption of life model* when using GPT queries and chunk length of 700 symbols. Generally, *text-embedding-3-small* performed best with the longest chunk lengths studied, showing a trend of increasing recall with the increase in chunk length. The topic *Strange/Unusual/Intense* had the lowest recall, possibly due to the uncertainty in topic specification.

For *Slovak-BERT* embedding indices, the recall values were slightly lower, but still impressive given the simplicity of this language model. Better results were achieved using GPT queries, with the best recall value of 47.1% for the topic *Radical adoption of life model* at a chunk length of 700 symbols, with embeddings created from chunks with removed stropwords. Generally, this embedding model benefited most from the stopwords removal preprocessing step.



These findings highlight the potential of leveraging LLMs for specialized domains like analysis of texts with religious themes.

As for *BGE M3* embeddings, the results were impressive, achieving high recall, though not as high as OpenAI's embeddings. However, considering that *BGE M3* is an open-source model, these results are remarkable.

These findings highlight the potential of leveraging LLMs for specialized domains like analysis of texts with religious themes. Future work could explore the connections between text chunks using clustering techniques with embeddings to discover hidden associations and inspirations of the text authors. For theologians, future work lies in examining the retrieved text parts to identify deviations from official teaching of Catholic Church, shedding light on movement's interpretations and insights.

ACKNOWLEDGEMENTS

Research results were obtained with the support of the Slovak National competence centre for HPC, the EuroCC 2 project and Slovak National Supercomputing Centre under grant agreement 101101903-EuroCC 2-DIGITAL-EUROHPC-JU-2022-NCC-01.

Computational resources were procured in the national project National competence centre for high performance computing (project code: 311070AKF2) funded by European Regional Development Fund, EU Structural Funds Informatization of society, Operational Program Integrated Infrastructure.

REFERENCES

[1] Matúš Pikuliak, Štefan Grivalský, Martin Konôpka, Miroslav Blšták, Martin Tamajka, Viktor Bachratý, Marián Šimko, Pavol Balážik, Michal Trnka, and Filip Uhlárik. Slovakbert: Slovak masked language model, 2021.

[2] Jianlv Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. Bge m3-embedding: Multi-lingual, multi-functionality, multi-granularity text embeddings through self-knowledge distillation, 2024.

[3] Liang Wang, Nan Yang, Xiaolong Huang, Linjun Yang, Rangan Majumder, and Furu Wei. Multil-ingual e5 text embeddings: A technical report, 2024.

[4] Harrison Chase. Langchain. https://github.com/langchain-ai/langchain, 2022. Accessed: May 2024.

[5] Xinbei Ma, Yeyun Gong, Pengcheng He, Hai Zhao, and Nan Duan. Query rewriting for retrieval-augmented large language models, 2023.

[6] Rolf Jagerman, Honglei Zhuang, Zhen Qin, Xuanhui Wang, and Michael Bendersky. Query expansion by prompting large language models, 2023.

03

HPC Popularization

Short News

All Hands Meeting in the High Tatras

In the beautiful setting of the High Tatras, members of the European National Competence Centers and Centers of Excellence gathered for the **EuroCC 2 & CoEs & CASTIEL 2 Intermediate Conference**. The event took place from April 22 and aimed to discuss project progress and share achieved successes.

The first day began with a warm welcome from the project management team and local organizers from the Slovak National Competence Center for HPC. Participants discussed recent developments and results from the first project evaluation. A key part of the discussions focused on synergies between the Competence Centers and Centers of Excellence, laying the foundation for joint activities and the exchange of experiences.

The second day focused on practical issues such as funding and strategic planning. Workshops centered on budgetary rules and discussions on NCC and CoE competencies provided valuable insights needed for sustainable growth. Participants also had the opportunity to take a tour around Štrbské Pleso lake, strengthening their informal bonds.

Regional Meeting of National Competence Centers for HPC in Central Europe

On June 10, the third meeting of the Central European Working Group of NCC for HPC took place in Grundlsee. The event was organized by NCC Austria and attended by representatives from Poland, Austria, Croatia, the Czech Republic, Slovakia, Slovenia, and Hungary.

The main topic of the meeting was the collaboration between NCCs and other institutions, as well as improving corporate profiles on LinkedIn. The opening speech was given by Markus Stöhr, head of the Austrian NCC for HPC, followed by presentations from each center. Discussions focused on enhancing training programs and optimizing LinkedIn profiles, with a workshop led by content marketing expert Natascha Trzepizur. The final session was dedicated to artificial intelligence.

eurocc.nscc.sk

NCC Slovakia & NCC Poland ORCA Hands-on Workshop

V On May 21-22, 2024, the ORCA hands-on workshop took place in Bratislava, organized by the national competence centers for HPC from Slovakia and Poland. The workshop focused on the software quantum-chemical package ORCA and its practical applications.

Participants were introduced to the basics and advanced techniques of working with ORCA. Lecturer Klemens Noga from Cyfronet presented the Polish HPC ecosystem and opportunities for users in computational chemistry. The program also included a tour of the Devana supercomputer at the Computing Center of the Slovak Academy of Sciences. The workshop provided participants with valuable knowledge for working with ORCA in quantum-chemical computations on HPC systems.

HPC Opportunities for SMEs

On March 19, an event titled "*Discover the Potential of Supercomputing in Practice*" was held in Bratislava, bringing together experts from the National Competence Center for HPC and the Slovak Chamber of Commerce and Industry. The goal was to introduce companies to the benefits and possibilities of using high-performance computing (HPC) in their business operations.

Participants had the opportunity to learn about modern technologies and their applications in development and manufacturing technology. Lucia Demovičová and Michal Pitoňák presented the NCC HPC services and discussed future collaboration opportunities. Companies could learn about testing allocations of HPC resources and involvement in free pilot projects, offering an attractive opportunity to adopt HPC without significant initial investments. The event also included a tour of the Devana supercomputer, providing a unique insight into the world of high-performance computing.

Popularization Lectures

The team from the National Competence Center for HPC organized a series of popularization lectures focused on current topics in science and technology. The lectures covered a variety of fascinating subjects, such as the role of high-performance computers and graphics processors in DNA genome analysis, conformational preferences of the naturally disordered tau protein, curious artificial intelligence, and an atomistic view of biopolymers. Other topics included modern technologies and challenges in automatic speech processing, modeling geoscientific data to understand Earth's geodynamic processes, quantum computing and its practical applications, and quantum complexity and simulations. The lectures provided participants with valuable insights and inspiration, contributing to a broader understanding of advanced scientific fields. We are preparing another series of popularization lectures and look forward to your participation!

Free IT courses

The National Competence Centre for High Performance Computing (HPC) provides the opportunity to learn new IT skills through free online courses.

For the winter semester, we have prepared a variety of courses focusing on areas such as artificial intelligence, machine learning, Python and Julia programming, interactive visualization, and database management. Among them are, for example *A Gentle Introduction to AI* or *Machine Learning in Python*.

Participants have the opportunity to use the Devana supercomputer for some hands-on exercises. We regularly update the courses based on feedback to reflect current needs and trends.

Sign up for one of our courses today!

Amidst the picturesque landscape of the Slovak High Tatra Mountains, the members of European National Competence Centres and Centres of Excellence met during the all-hands meeting to discuss the project development and show the progress that has been made.

Superheroes 4 Science

Visegrad Fund

Superheroes 4 Science, supported by the Visegrad Fund, focuses on promoting High Performance Computing (HPC) and supercomputing, which are crucial for the development of all scientific disciplines. HPC and supercomputing are often considered the third pillar of science, alongside theory and experiment. Therefore, it is essential to popularize the importance of supercomputers and their applications, which have a positive impact on everyday life.

In the ongoing project, we are now focusing on explaining High Performance Computing (HPC) and its applications in areas such as artificial intelligence (AI) and quantum computing (QC). HPC is a fundamental area of technology that uses supercomputers to solve complex problems and perform computationally intensive tasks. AI and QC are particularly challenging, but their significance for the future is immense.

Project Partners

Czech Republic
IT4Innovations National Supercomputing Center at VŠB – Technical University of Ostrava is a leading center in the fields of HPC, data analysis, AI, and QC.

Poland
The Poznań Supercomputing and Networking Center (PSNC) provides comprehensive services in the fields of cloud computing and HPC, and operates the supercomputer Altair.

Slovakia
The Slovak National Supercomputing Center (NSCC) supports all HPC activities in Slovakia and operates the supercomputer Devana.

Goals and Activities

The project aims to educate the general public and inspire the younger generation to pursue studies in scientific and technical fields. The project includes interactive educational materials and comics that explain in simple terms how high-performance technology and artificial intelligence can improve and enhance our lives. We also plan to create a supercomputer game that will introduce these technologies to younger generations in a fun and engaging way.

Superheroes 4 Science is an example of how international collaboration and support for innovative technologies can shape the future of science and technology. The second phase of the project (2023 – 2025) aims to expand awareness of HPC, AI, and QC, and to inspire the next generation of scientists and technicians.

hpc focus

2024

EDITOR-IN-CHIEF
Lucia Demovičová

GRAPHIC DESIGN & DTP
Gabriela Obadalová

PHOTOS
Pavol Novák
Shutterstock

PERIODICITY
Yearly

PUBLISHER
National Supercomputing Centre
Dúbravská cesta 3484/9
841 04 Bratislava
Slovak Republic
Tel.: +421 904 816 609
eurocc@nscc.sk
eurocc.nscc.sk

in cooperation with

Centre of Operations of SAS
Computing Centre of SAS
Dúbravská cesta 9
845 35 Bratislava
Slovak Republic
Tel.: +421 (0)2/ 3229 3111
vssav@savba.sk
vs.sav.sk

ISBN 978-80-89871-25-4
ISSN 2729-9090

The texts have not undergone proofreading.

National Supercomputing
Centre

Dúbravská cesta 3484/9
841 04 Bratislava
Slovak Republic

eurocc.nscc.sk

ISBN 978-80-89871-25-4
ISSN 2729-9090