Accelerating Metagenomic Identification of DNA Sequences Using Artificial Neural Networks

Patryk Filip Gryz[1*] and Robert Nowak[1*]

[1*]Institute of Computer Science, Warsaw University of Technology, Nowowiejska 15/19, Warsaw, 00-665, Poland.

*Corresponding author(s). E-mail(s): patryk.gryz.stud@pw.edu.pl; robert.nowak@pw.edu.pl;

Abstract

The rapid increase in the availability of DNA sequence data necessitates efficient and scalable methods for metagenomic identification. Traditional tools like BLAST, although accurate, often require significant computational resources and time, limiting their applicability to large datasets. Recent advances in artificial neural networks (ANNs) demonstrate their potential to address this issue by offering faster and more accurate solutions.

This study introduces a novel ANN-based approach for the efficient metagenomic identification of DNA sequences. The proposed method reduces the complexity of large datasets by leveraging contrastive learning within an ANN to select representative sequences. This approach enhances computational efficiency by focusing classification efforts on these representatives rather than on the entire dataset. Comparative experiments were conducted on a benchmark dataset, evaluating the proposed ANN method alongside classical techniques such as k-mer embeddings and the Needleman-Wunsch algorithm.

Experimental results revealed that the ANN-based approach achieves the highest weighted average classification quality among all tested methods, outperforming both classical approaches. Additionally, the ANN method demonstrates competitive execution times, benefiting from parallelized computations enabled by GPU processing.

In conclusion, the study highlights the potential of ANN-based methods to bridge the gap between precision and scalability in genetic taxonomy. We developed Exquisitor, a command-line tool implementing proposed method.

Future research directions include enhancing the network's flexibility for varying sequence lengths and exploring its application for dimensionality reduction in broader bioinformatics tasks.

Keywords: DNA sequencing, metagenomic identification, artificial neural networks, contrastive learning, sequence clustering

1 Background

The use of information about genetic material found in the environment, combined with sequence databases, enables metagenomic identification – the identification of organisms present in a given sample. With the development of next-generation sequencing methods[1], sequencing costs have decreased significantly, and the number of DNA sequences being analyzed has increased[2]. The time required for metagenomic identification using traditional tools, such as BLAST, is unacceptable in many applications – can span several hours, even for relatively small collections of analyzed sequences (for example 4096 sequences), due to huge size and explosive growth of sequencing databases [3]. This process require significant computational resources. It is necessary to create new methods to allow analyzing new samples of data in acceptable time using available resources.

The primary objective of this study is to propose fast and high quality metagenomic identification method. We propose usage an artificial neural network to support the selection of representative genetic sequences. Instead of classifying entire environmental genetic sequences sample, the proposed approach focuses on identifying a subset of sequences that best represent the sample. These selected representatives are then used to identification, reducing computational complexity while maintaining accuracy.

Historically, one of the first algorithms that enabled metagenomic identification was the Needleman-Wunsch algorithm [4], developed in 1970, which allowed for the comparison of genetic sequences. However, the first solution that enabled this task within a reasonable time frame was a tool created in 1983 by D. Lipman and W. Wilbur [5]. Their method was based on dividing sequences into k-tuples, a generalization of k-mers, and comparing them. An advancement of this approach was the BLAST algorithm, introduced in 1990 [6], which also relied on k-mers and allowed for the efficient search of similar sequences in sequence databases.

The metagenomic ideitification became popular over the past 15 years, due to sequence database growth. A notable example of interesting tool is the development of the MetaPhlAn tool in 2012 [7], which uses marker genes to compare the species composition of metagenomic samples. Another marker gene-based approach is mOTUs2 [8], which identifies and analyzes markers unique to specific microbial strains.

A different technique was employed in Centrifuge [9], developed in 2016, which utilizes sequence indexing based on the Burrows-Wheeler transform [10] and the Ferragina-Manzini index [11] for efficient sequences retrieval. A more classical approach is used in Kraken [12], introduced in 2014, which combines k-mers with an indexed sequence database. An unconventional method for metageonomic identification was proposed in 2022 with BERTax [13], which applies a transformer model [14] to DNA sequence analysis, treating DNA as a specialized language and allowing for classification without the need for reference databases. Another machine learning-based

solution is CGRclust [15], which clusters DNA sequences using a two-dimensional chaos game representation and integrates unsupervised contrastive learning with convolutional neural networks.

Currently we can propose three types of methods developed for metagenomic identification: methods using alignment, methods based on k-mers and methods using marker genes. Alignment-based tools should yield the best results, but they require significant computational resources. Algorithms based on k-mers are very fast, but they do not take into account all the information about the structure of the input genetic sequences, which reduces their quality. Solutions using marker genes are characterized by good quality, but they allow analysis only for subsets of organisms for which marker genes are known.

Although recent years have seen studies on metagenomic identification using machine learning methods, there is still a lack of research focusing on the application of artificial neural networks with contrastive learning for efficient sequence clustering. In many fields, artificial neural networks (ANN) have demonstrated the ability to outperform classical heuristic algorithms, offering faster and more accurate results. By leveraging the capabilities of ANN, this works aims to accelerate metagenomic identification process while maintaining at least the same level of quality as traditional methods. Such an approach could significantly accelerate the classification process using available tools.

The rest of the paper is organized as follows. Section 2 defines the problem, presents the proposed approach, and provides implementation details of developed tool Exquisitor. Section 3 describes the experimental setup, including used datasets, the performance metrics for evaluation, and the results obtained from experiments with analysis. Finally, section 4 concludes the paper by summarizing the main findings, discuss on the limtiations of proposed approach and suggesting potential directions for future research.

2 Methods

The core component of proposed tool, Exquisitor is a processing pipeline designed with flexibility in mind, allowing the integration of various DNA sequence clustering methods. The pipeline consists of a series of steps executed sequentially to perform metagenomic identification. A schematic representation of the pipeline is show in Figure 1, where input and output data are represented by circles, and each processing step is represented by a rectangle. The pipeline is composed of the following five steps:

1. **Preparation of DNA Sequences**
   This step is responsible for verifying the correctness of the provided DNA sequences and aligning them to a required length, which may be necessary for subsequent processing.
2. **Dissimilarity Calculation**
   In this step, the dissimilarity between DNA sequences is calculated using one of three available algorithms: two classical methods and one custom-developed approach. The resulting dissimilarity matrix serves as the basis for clustering.
3. **Sequence Clustering**

Based on the previously computed dissimilarities, sequences are grouped into clusters. Each cluster consists of one representative sequence and several member sequences. The main objective of this step is to reduce the number of sequences processed in later stages by selecting the most informative representatives.

4. **Database Search**
   In this step, the representative sequences are queried against a reference database to identify similar sequences. This allows the system to infer the likely origin of the sequences and determine the organisms they may belong to.
5. **Postprocessing**
   In this final step, information from the clusters is integrated with the database search results.

2.1 Exquisitor: Method Description

The proposed method involves reducing the dimensionality of input DNA sequences to a feature vector in the form of $\mathbb{R}^{64}$ using an artificial neural network. The neural network employs contrastive learning, which enables the learning of representations while preserving dissimilarity between sequences. The dissimilarity between sequence representations will be calculated using cosine dissimilarity, expressed by the formula in equation (1).

$$dsim(A, B) = 1 - sim(A, B) \tag{1}$$

where:

$$sim(A, B) = \cos\theta = \frac{A \cdot B}{\|A\|\|B\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \cdot \sqrt{\sum_{i=1}^{n} B_i^2}}$$

$A, B-$compared vectors,

$\theta-$angle between A and B,

A_j, B_j-j-th element of the vector A and B, respectively.

2.1.1 Artificial Neural Network

Architecture

The ANN model consists of two parts. The first part comprises two convolutional blocks with batch normalization and is responsible for feature extraction from the sequence. The second part utilizes fully connected layers with dropout and the GELU activation function [16] and is connected to the first part of the model via a flattening layer.

The model output is the output of the final linear layer, which is a feature vector of dimension $\mathbb{R}^{64}$.

Schematically, the architecture is presented in Figure 2.

Input

The input to the model consists of DNA sequences of length 150, which are encoded into a vector of dimensions 1×600 using one-hot encoding [17].

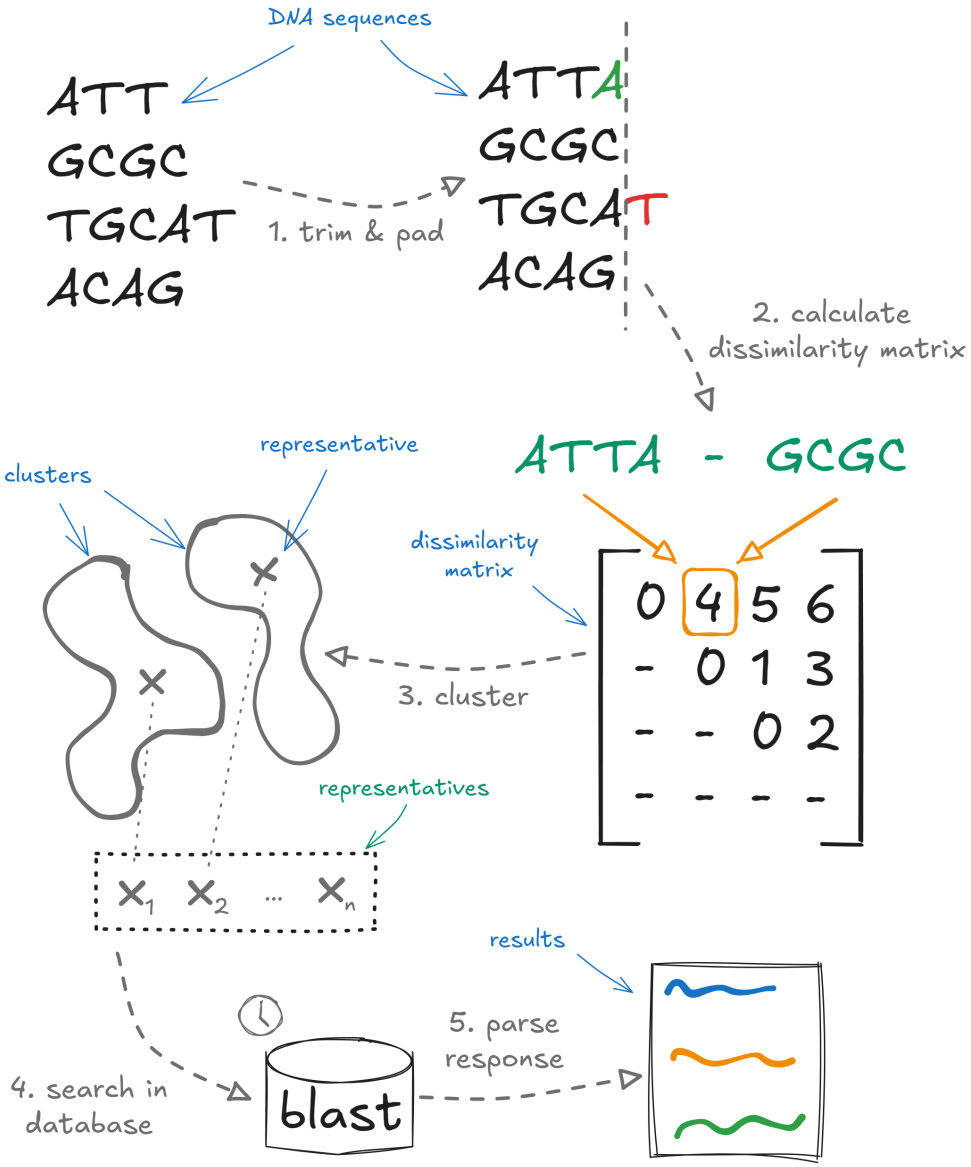

Fig. 1 Diagram of the pipeline.

Training Examples

The training and validation examples consist of an anchor, a positive sequence that is similar to the anchor, and a negative sequence that is dissimilar to the anchor.

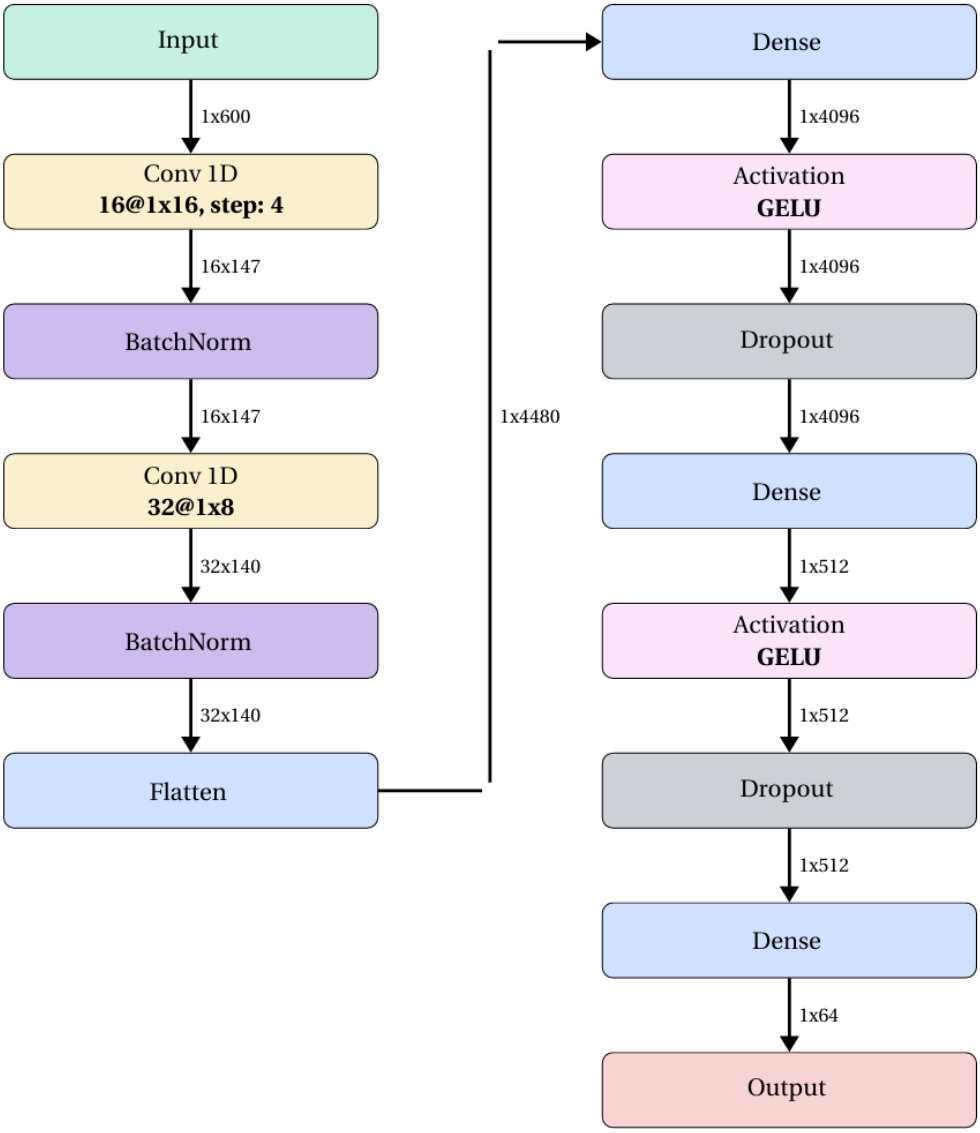

Fig. 2 Diagram of ANN architecture.

Dataset

The training and validation datasets were created based on the first sample of genetic sequences from the CAMI II Toy Human Microbiome Project dataset [18]. The sample contains simulated metagenomic data from the human skin microbiome. Examples were obtained by randomly selecting anchors from the dataset and modifying these anchors to create positive and negative sequences. The modification involved a point

mutation of a given nucleotide to another one. In the case of positive sequences, the change affected between 0% and 20% of the anchor length, whereas for negative sequences, the change ranged from 20% to 80% of the anchor length.

Loss function

The loss function used is defined as:

$$\text{Contrastive loss} = [m_{pos} - s_{pos}]_+ + [s_{neg} - m_{neg}]_+ \tag{2}$$

where:

m_{pos} – similarity margin between the positive sequence and the anchor,
m_{neg} – similarity margin between the negative sequence and the anchor,
s_{pos} – cosine similarity of the positive sequence to the anchor,
s_{pos} – cosine similarity of the negative sequence to the anchor.

Learning

The model was trained on a dataset of 10^6 training examples and 10^4 validation examples. The optimizer used was $AdamW$[19].

Quality

As a measure of model quality, the contrastive loss of the model was computed on the validation set.

Parameters

Experiments were conducted to determine the optimal training parameters for the artificial neural network model. The tested parameters included the learning rate coefficient λ, weight decay w, the coefficient γ used in the exponential learning rate decay, the dropout rate, and the effectiveness of using three perceptron layers.

As a result of these experiments, the best parameters were selected:

$$\lambda = 10^{-6}, \quad w =, \quad \gamma = 0.99999$$

a dropout rate of 0.5, and a confirmed positive impact of using three perceptron layers.

Additionally, the loss function was used with the following parameters:

$$m_{pos} = 1.0, \quad m_{neg} = 0.25.$$

Results of Learning

As a result of training the neural network, the model achieved a validation set loss of 0.17.

2.2 Exquisitor: Implementation Details

The solution consists of two components. The first is a library that contains components enabling the creation of pipelines for genetic material analysis in the form of metagenomic identification. The second component is a console application, which uses the library's components to allow for the creation and configuration of processing pipelines. Additionally, a web application was created, which allows users to submit analysis tasks through a web browser.

The console application was implemented using the *clap* library[20]. In library the k-medoid algorithm was provided by the *kmedoids* library[21], and the neural network was built using the *burn* library[22] with the *wgpu* execution environment. Web application was implemented using the *axum* library[23] with *tokio*[23] async environment.

Additionally, the following tools were used in the work:

- *cargo* as the package manager and build system in Rust;
- *rustup* for automatic management of Rust versions;
- *clippy* for static code analysis in Rust;
- *rustfmt* for automatic formatting of Rust source code;
- *cargo test* for conducting unit tests;
- *git* as the version control system, enabling change tracking and code history management.

3 Results

This sections presents the results of the experiments conducted to evaluate the proposed solution (Exquisitor), which aims to improve performance while maintaining the quality of metagenomic identifacation. The study involved comparing the new approach with two classical methods using the same dataset. The evaluation focused on execution time and identification quality to assess the effectiveness of the proposed approach.

3.1 Datasets

3.1.1 Dataset Description

In the experiments, the *CAMI II Toy Human Microbiome Project*[18] dataset was used, which is the same dataset used for training the artificial neural network model. Dataset was chosen, because it was created for benchmarking bioinformatics tools and contains a large number of sequences, enabling its use both in experiments and in the learning process.

Subsets of sequences were created from the dataset with sizes expressed by the formula 2^k for $k \in [0, 12]$. Subsets are disjoint, and only those sequences that were not used for training the ANN model were employed in their construction.

3.1.2 Dataset Preparation

The subsets were created by randomly sampling, without replacement, the indices of genetic sequences from the reference dataset that were to be included in each subset. Indices of sequences used in the training and validation sets of the ANN were excluded from the sampling.

3.2 Base-line Algorithms

Two classical approaches were implemented and used for comparison with the proposed neural network.

3.2.1 Modified Needleman-Wunsch Algorithm

The first classical approach is the Needleman-Wunsch algorithm. This algorithm allows for global alignment of genetic sequences. It uses a similarity matrix between sequences, which is constructed according to the rules given in equation (3). A modification of the approach presented in equation (4) is applied in the work.

$$\begin{aligned} D_{i,0} &= i \cdot g, && \text{for } i \in [1, n+1] \\ D_{0,j} &= j \cdot g, && \text{for } j \in [2, m+1] \\ D_{i,j} &= \max \begin{cases} D_{i-1,j} + g \\ D_{i,j-1} + g \\ D_{i-1,j-1} + s(A_i, B_j) \end{cases}, && \text{for } i \in (1, n+1] \text{ and } j \in (1, m+1] \end{aligned} \tag{3}$$

where:

$$\begin{aligned} A, B &- \text{compared sequences}, \\ n, m &- \text{lengths of sequences } A \text{ and } B, \\ D &- \text{similarity matrix of size } n+1 \text{ x } m+1, \\ g \in \mathbb{R} &- \text{penalty for a gap}, \\ s(A_i, B_j) \in \mathbb{R} &- \text{similarity between the } i\text{-th element in sequence A} \\ &\quad \text{and the } j\text{-th element in sequence B}. \end{aligned}$$

$$D_{i,j} = \min \begin{cases} D_{i-1,j} + g \\ D_{i,j-1} + g \\ D_{i-1,j-1} + s(A_{i-1}, B_{j-1}) \end{cases}, \quad \text{for } i \in (1, n+1] \text{ and } j \in (1, m+1] \tag{4}$$

In the solution, the following parameter values were adopted:

$$g = 2,$$

$$s(a,b) = \begin{cases} 0, & \text{for } a = b, \\ 1, & \text{for } a \neq b. \end{cases}$$

3.2.2 *k*-mer Embeddings

The second approach will be the use of k-mers as embeddings, which allows for the representation of DNA sequences as numerical vectors. These vectors will then be compared using Euclidean distance.

3.3 Experiments

3.3.1 Execution Time of Metagenomic Identification with Exquisitor

Objective

Measuring the execution time of metagenomic identification using different genetic sequence clustering methods.

Assumptions

1. Sequences clustering is deterministic.
2. Metagenomic identification is the only task running on the machine.

Results

As a result of the experiment, the execution time for metagenomic identification was obtained for all methods. In the case of the method using the modified Needleman-Wunsch algorithm, results for metagenomic identification of 4096 sequences could not be obtained due to exceeding the execution time. Figure 3 shows a graph of the metagenomic identification execution time as a function of the number of input sequences for: the method with the modified Needleman-Wunsch algorithm (NW), the method using k-mer embeddings (k-mer), the method using artificial neural networks (ANN), and for metagenomic identification of all sequences without using a pipeline (NP). Detailed execution times are provided in Table 1.

Conclusions

The method using the modified Needleman-Wunsch algorithm exhibited the fastest increase in execution time for metagenomic identification. The sharp rise is due to the time-consuming process of comparing sequences with each other. Despite reducing the number of sequences being classified, the execution time using this method exceeded that of metagenomic identification for all sequences.

The methods using k-mer embeddings and ANN showed comparable execution times, with a slight advantage for the former. Both methods use sequence representations for comparison, resulting in a slower increase in the time needed to build the dissimilarity matrix. These methods performed faster than metagenomic identification for all sequences. In the case of the ANN, the shorter execution time for 4096 sequences could be due to the simultaneous computation of embeddings for all sequences using the GPU.

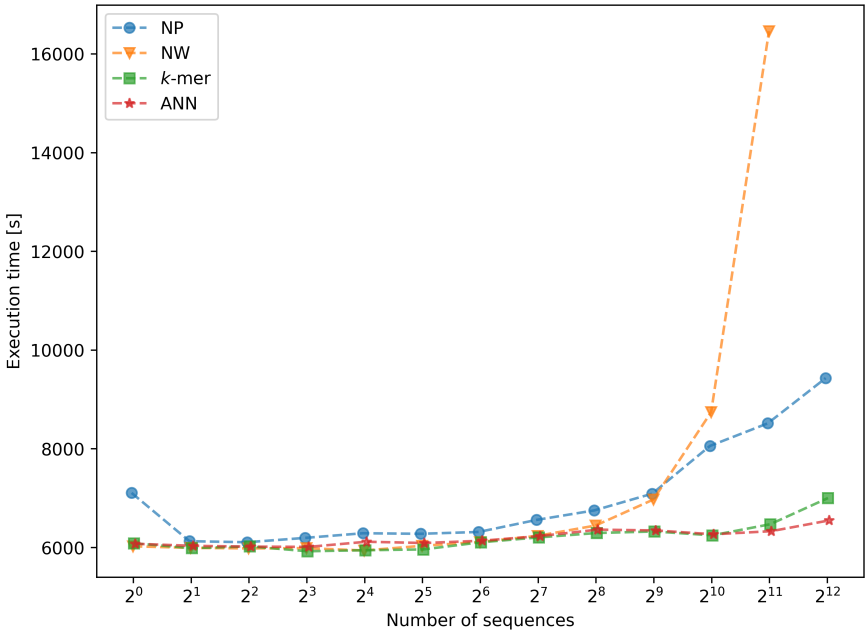

Fig. 3 Metagenomic identification execution time.

3.3.2 Quality of Metagenomic Identification with Exquisitor

Objective

Examining the quality of metagenomic identification using the implemented methods in comparison to the metagenomic identification of all sequences.

Assumptions

1. Sequences clustering is deterministic.
2. `BLASTn` is deterministic and always returns the same results for a given sequence and specified parameters.
3. In the case where it is not possible to compute the quality for a given metagenomic identification run, a quality value of 0 is assumed. This value is not considered when calculating the weighted average quality.

3.4 Exquisitor Performance Measures

Quality of metagenomic identification was measured using a modified Jaccard index, expressed by equation 5. The measure was used to compare the quality of metagenomic identification performed using the implemented methods against metagenomic identification without the use of a processing pipeline.

Table 1 Metagenomic identification execution time.

Number of sequences	Execution time [s]			
	NP	**NW**	***k*-mer**	**ANN**
1	7107	**6025**	6087	6077
2	6129	5994	**5988**	6035
4	6108	**5983**	6024	6019
8	6196	5988	**5925**	6014
16	6290	**5941**	5946	6118
32	6279	6035	**5962**	6092
64	6316	6113	**6107**	6139
128	6560	6238	**6206**	6233
256	6753	6446	**6295**	6363
512	7091	6972	**6326**	6347
1024	8059	8743	**6248**	6269
2048	8517	16461	6472	**6332**
4096	9433	**timeout**	7003	6550

$$\mathrm{Q} = \frac{\sum_{r \in (O(R) \cap O(E))} R_r}{\sum_{r \in (O(R) \cup O(E))} R_r} \tag{5}$$

where:

R – set of reference results,
E – set of obtained results,
R_r – number of results in reference set assigned to the organism r
$O(X)$ – set of unique organisms, for which results were assigned in the set X

To compare the quality of multiple metagenomic identification runs, the weighted average quality, defined by equation 6, was used.

$$Q_{\mathrm{avg}} = \sum_{c \in C} \frac{n_c}{n} Q_c \tag{6}$$

where:

C – set of metagenomic identification runs,
Q_c – quality of metagenomic identificationn c,
n_c – number of input sequences for metagenomic identification c,
n – number of input sequences $n = \sum_{c \in C} n_c$.

Results

The quality of the classification was calculated using the measure expressed by equation 5 for the implemented methods in comparison to the full metagenomic identification. Figure 4 shows a graph of classification quality as a function of the

number of input sequences, with detailed quality results provided in Table 2. The weighted average quality calculated using equation 6 for the method with the modified Needleman-Wunsch algorithm was 0.899, for the k-mer embedding method it was 0.921, and for the method using the artificial neural network, it was 0.944.

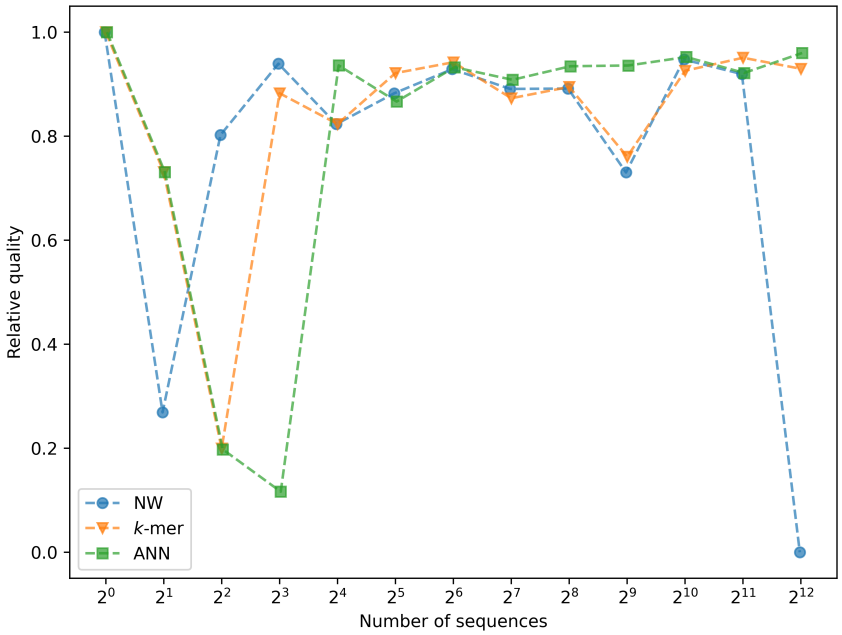

Fig. 4 Metagenomic identification quality.

Conclusions

All implemented methods achieved very good classification quality results, exceeding 0.7 in cases where the number of groups was significantly larger than the number of input sequences. The weighted average quality for all methods also reached a high value.

The method using the artificial neural network achieved the best weighted average quality, slightly outperforming the k-mer embedding method and the method with the modified Needleman-Wunsch algorithm. The artificial neural network method owes its result to the model's consideration of the full sequence structure, which allowed for a better determination of dissimilarity between sequences.

Table 2 Metagenomic identificationn quality.

Number of sequences	Method		
	NW	***k*-mer**	**ANN**
1	**1.0**	**1.0**	**1.0**
2	0.27	**0.73**	**0.73**
4	**0.8**	0.2	0.2
8	**0.94**	0.88	0.12
16	0.82	0.82	**0.94**
32	0.88	**0.92**	0.87
64	0.93	**0.94**	0.93
128	0.89	0.87	**0.91**
256	0.89	0.89	**0.93**
512	0.73	0.76	**0.94**
1024	**0.95**	0.93	**0.95**
2048	0.92	**0.95**	0.92
4096	0.0	0.93	**0.96**

4 Discussion

4.1 Interpretation of Results

The results of the experiments partially confirmed the expectations set at the beginning of the research. All methods achieved satisfactory metagenomic identification quality. The method utilizing the artificial neural network performed particularly well, achieving the best weighted average classification quality while maintaining execution time comparable to the method based on k-mer embeddings. The low metagenomic identification quality with a small number of input sequences may be due to the inappropriate selection of group representatives, caused by too few available sequences. Therefore, the implemented methods should only be used in cases where the number of input sequences exceeds a certain threshold, and their use significantly reduces the time required for the metagenomic identification process. For the data tested, this condition is met for 128 sequences, where the metagenomic identification quality was no less than 0.87, and the use of any grouping method reduced the time by at least 5 minutes, resulting in a 5% acceleration compared to the metagenomic identification of all sequences. The use of the artificial neural network allowed for an improvement in classification quality compared to other methods. In addition to achieving good classification quality, it does not lag behind classical methods.

4.2 Limitations

4.2.1 Experiments Duration

The full process of metagenomic identification of DNA sequences for each method and each experimental subset took approximately 5 days, which led to the limitation of the number of experiments to single run and subset size to 4096 sequences.

4.2.2 Single Dataset

In work, only one sample from the chosen dataset was used. The selection of a sample containing DNA sequences from the human skin microbiome might have limited the space of analyzed sequences, which could have influenced the experimental results, as the both the artificial neural network and the experiments used DNA sequences from the same microbiome.

4.2.3 Single Dissimilarity Matrix

The use of a single dissimilarity matrix for all sequences limits the number of input sequences, as the matrix build time of matrix scales quadratically with the number of sequences.

4.3 Possible Improvements

4.3.1 Architecture of Artificial Neural Network

The proposed artificial neural network architecture is not flexible and is only suitable for analyzing sequences of similar lengths. It would be possible to modify the model's architecture to incorporate recurrent neural networks (RNNs) and transformers, enabling the processing of sequences with varying lengths. RNNs and transformers could replace the first part of the model, which currently relies on convolutional layers. They would be used to create embeddings, which would later be processed by linear layers to generate an embedding that preserves dissimilarity properties.

4.3.2 Batching Sequences

Creating a single dissimilarity matrix for very large samples is not optimal, as it significantly impacts the performance of the method. To address this issue, pre-clustering the sequences into batches can provide a solution. Processing in batches may reduce quality but allows for maintaining a short execution time.

4.3.3 Automatated Determination Of The Number of Groups

Current approach takes the number of created groups as a parameter, which can influence the quality of the results. For very similar sequences, a large number of groups is excessive and does not improve the results. Conversely, for highly diverse sequences, a low number of groups can reduce quality. Automating the determination of the optimal number of groups could solve the issue and minimize the need for fine-tuning

5 Conclusions

5.1 Summary of Findings

The experiments demonstrated that the proposed method based on artificial neural networks outperforms traditional approaches in terms of quality, while maintaining

execution times comparable to those of the classical k-mer method. This highlights the potential of neural networks for improving performance without significant computational overhead.

5.2 Future Research Directions

Future research could focus on utilizing artificial neural networks as an independent tool for metagenomic identification. Another promising direction is the application of artificial neural networks for dimensionality reduction of genetic sequences, which may improve efficiency and performance in various analyses. Planned development efforts are focused on providing interfaces for the Exquisitor tool, enabling its integration with existing popular metagenomic pipelines.

Code availability

Exquisitor is distributed under the terms of both the MIT license and the Apache License (Version 2.0). The source code is available at https://github.com/pfgryz/exquisitor.

Author contributions

P.G and R.N. identified the problem, P.G. designed the approach, downloaded the data, implemented the software, performed numerical experiments, P.G. and R.N. prepared the draft. All authors have read and agreed to the published version of the manuscript.

Funding

A statutory Research Grant from the Institute of Computer Science, Warsaw University of Technology, supports this work.

References

[1] Reinartz J, Bruyns E, Lin JZ, Burcham T, Brenner S, Bowen B, et al. Massively parallel signature sequencing (MPSS) as a tool for in-depth quantitative gene expression profiling in all organisms. Briefings in Functional Genomics. 2002;1:95–104.

[2] Muir P, Li S, Lou S, Wang D, Spakowicz DJ, Salichos L, et al. The real cost of sequencing: scaling computation to keep pace with data generation. Genome Biology. 2016;17:53.

[3] Kodama Y, Shumway M, Leinonen obotINSDC Rasko. The sequence read archive: explosive growth of sequencing data. Nucleic Acids Research. 2011 10;40(D1):D54–D56. https://doi.org/10.1093/nar/gkr854. https://academic.oup.com/nar/article-pdf/40/D1/D54/9482007/gkr854.pdf.

[4] Needleman SB, Wunsch CD. A general method applicable to the search for similarities in the amino acid sequence of two proteins. Journal of Molecular Biology. 1970;48(3):443–453.

[5] Wilbur WJ, Lipman DJ. Rapid similarity searches of nucleic acid and protein data banks. Proceedings of the National Academy of Sciences. 1983;80:726–730.

[6] Altschul S, Gish W, Miller W, Myers E, Lipman D. Basic local alignment search tool. Molecular Biology. 1990;215:403–410.

[7] Segata N, Waldron L, Ballarini A, in. Metagenomic microbial community profiling using unique clade-specific marker genes. Nature Methods. 2012;9:811–814.

[8] Milanese A, Mende DR, Paoli L, in. Microbial abundance, activity and population genomic profiling with mOTUs2. Nature Communications. 2019;10:1014.

[9] Kim D, Song L, Breitwieser FP, L SS. Centrifuge: rapid and sensitive classification of metagenomic sequences. Genome Research. 2016;26:1721–1729.

[10] Burrows M, Wheeler DJ. A Block-sorting Lossless Data Compression Algorithm; 1994. .

[11] Ferragina P, Manzini G. Opportunistic data structures with applications. In: Proceedings 41st Annual Symposium on Foundations of Computer Science; 2000. p. 390–398.

[12] Wood DE, Salzberg SL. Kraken: ultrafast metagenomic sequence classification using exact alignments. Genome Biology. 2014;15:R46.

[13] Mock F Florian AMD Kretschmer, Kriese A, Böcker S, Marz M. Taxonomic classification of DNA sequences beyond sequence similarity using deep neural networks. Proceedings of the National Academy of Sciences. 2022;119:e2122636119.

[14] Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, et al. Attention is All you Need. In: Guyon I, Luxburg UV, Bengio S, Wallach H, Fergus R, Vishwanathan S, et al., editors. Advances in Neural Information Processing Systems. vol. 30. Curran Associates, Inc.; 2017. .

[15] Alipour F, Hill KA, Kari L. CGRclust: Chaos Game Representation for twin contrastive clustering of unlabelled DNA sequences. BMC Genomics. 2024;25:1214.

[16] Hendrycks D, Gimpel K. Gaussian Error Linear Units (GELUs). arXiv: Learning. 2016;.

[17] Harris D, Harris S. Digital Design and Computer Architecture: From Gates to Processors. 1st ed. Burlington: Elsevier Science & Technology; 2007.

[18] Fritz A, Hofmann P, Majda S, in. CAMISIM: simulating metagenomes and microbial communities. Microbiome. 2019;7:17.

[19] Loshchilov I, Hutter F. Decoupled Weight Decay Regularization. In: International Conference on Learning Representations; 2017. .

[20] : clap. Available from: https://crates.io/crates/clap.

[21] Schubert E, Lenssen L. Fast k-medoids Clustering in Rust and Python. Journal of Open Source Software. 2022;7(75):4183.

[22] : burn. Available from: https://crates.io/crates/burn.

[23] : Axum. Available from: https://crates.io/crates/axum.