SOFTWARE

Open Access

NERVE 2.0: boosting the new enhanced reverse vaccinology environment via artificial intelligence and a user-friendly web interface

Andrea Conte[1†], Nicola Gulmini[1†], Francesco Costa[1,2†], Matteo Cartura[1], Felix Bröhl[3], Francesco Patanè[1] and Francesco Filippini[1*]

[†]Andrea Conte, Nicola Gulmini and Francesco Costa have contributed equally to this work.

*Correspondence: francesco.filippini@unipd.it

[1] Synthetic Biology and Biotechnology Unit, Department of Biology, University of Padua, Padua, Italy
[2] EMBL-European Bioinformatics Institute (EMBL-EBI), Hinxton, Cambridge, UK
[3] Bröhl Software Development, Bonn, Germany

Abstract

Background: Vaccines development in this millennium started by the milestone work on *Neisseriameningitidis B*, reporting the invention of Reverse Vaccinology (RV), which allows to identify vaccine candidates (VCs) by screening bacterial pathogens genome or proteome through computational analyses. When NERVE (New Enhanced RV Environment), the first RV software integrating tools to perform the selection of VCs, was released, it prompted further development in the field. However, the problem-solving potential of most, if not all, RV programs is still largely unexploited by experimental vaccinologists that impaired by somehow difficult interfaces, requiring bioinformatic skills.

Results: We report here on the development and release of NERVE 2.0 (available at: https://nerve-bio.org) which keeps the original integrative and modular approach of NERVE, while showing higher predictive performance than its previous version and other web-RV programs (Vaxign and Vaxijen). We renewed some of its modules and added innovative ones, such as *Loop-Razor*, to recover fragments of promising vaccine candidates or *Epitope Prediction* for the epitope prediction binding affinities and population coverage. Along with two newly built AI (Artificial Intelligence)-based models: *ESPAAN* and *Virulent*. To improve user-friendliness, NERVE was shifted to a tutored, web-based interface, with a noSQL-database to consent the user to submit, obtain and retrieve analysis results at any moment.

Conclusions: With its redesigned and updated environment, NERVE 2.0 allows customisable and refinable bacterial protein vaccine analyses to all different kinds of users.

Keywords: Reverse vaccinology, Vaccine candidates, Modular software, User-friendly website, Machine learning, Artificial intelligence

Background

In the early 2000s, the growing availability of genomic data and the development of more performing bioinformatics tools led to a revolution in vaccinology, i.e., to the birth of *Reverse Vaccinology* (RV) [1]. Starting with the work on *Neisseria meningitidis group B* by Rino Rappuoli's team [2], RV has enhanced the capacity to identify VCs by replacing several experimental tasks. This is possible via in-silico prediction steps on the genome and/or proteome of the pathogen of interest, with consequent time and cost benefits. Afterwards, RV has been applied to other pathogenic bacterial species. However, the first bioinformatics-driven approaches were pathogen-tailored and poorly generalizable [3]. To tackle the need to standardise VCs search process, the first published open-source RV platform was NERVE (*New Enhanced Reverse Vaccinology Environment)* [4], only for Linux users. It is based on a modular structure, and it extracts relevant information from a given pathogen proteome through six analytical steps, comprehending different bioinformatic tools (see Additional Fig. 1). These data, saved in a MySQL table, are then used to infer the presence of protein vaccine candidates (PVCs), which are collected in an HTML table.

Inspired by NERVE, further RV applications have been developed, including new and updated tools to improve VCs identification [3]. According to their working structure, RV programs can be classified into these two categories [5]: filter-based and machine learning (ML)-based. In the first ones, protein sequences are analysed by different tools to obtain useful information, such as their probability of being an adhesin or their subcellular localization. Then, this is passed to a decision tree that uses a priori cut-offs to select PVCs. NERVE [4], Jenner-predict [6], VacSol [7] and Vaxign [8] fall into this category.

In the second category, some specific protein sequence features are extracted and directly fed to a ML model. These features may include physicochemical descriptors, such as amino acid composition or hydrophobicity propensity. Next, an artificial neural network (ANN) is often adopted to classify input proteins in PVCs and not-PVCs, solving a common binary classification task. Examples of these programs are VaxiJen [9] and Bowman–Heinson [10, 11].

Differently from the cited ones, ReVac [12] uses a scoring system that ranks all PVCs from the most to the least likely one. It accepts bacterial proteomes and genomes as inputs, which are analysed by several bio-tools integrated in the Ergatis platform [13] and grouped according to the analysed feature, enabling parallel computations. This redundancy, as the authors specify, leads to more confident predictions. Nevertheless, the complex structure of ReVac represents a drawback because it requires unclear software dependency installation and scarce documentation, representing a major disadvantage for most of the users.

Despite recent and remarkable improvements in RV applications, such as the evaluation of virulence factors as PVCs [7] or population coverage predictions of pathogen input proteins [8], one of their main limitations remains the difficulty of installation and use. Most of the cited programs are not readily accessible because their installation procedures are often challenging due to unclear instructions or little support by the developers. Instead, accessibility to a broad plethora of users should be an essential goal of RV applications [5]. NERVE 1.0 installation was rather arduous as well because of the multiple dependencies which required manual downloading and configuration, and it is no longer supported as most of its Perl libraries are now obsolete.

We upgraded NERVE to tackle the usability and accessibility pitfalls common to many RV programs. We renewed most modules (also named components) and included new AI (Artificial Intelligence) ones. In the realm of vaccine development, advanced computational pipelines are now utilising AI systems, which provide great improvements in terms of accuracy and speed allowing also a better comprehensive analysis of entire proteomes, compared to non-AI methods [14].

Implementation

NERVE 1.0 was meant to be an environment, gathering different tools and making them collaborate to obtain a final PVCs ranking. The same approach is maintained in version 2.0.

The first major difference, compared to the original version, is the programming language: Perl was replaced by Python, especially for libraries availability, e.g. TensorFlow [15] for the modules using deep learning models. Regarding the input, FASTA files can be automatically retrieved from UniProt [16], in addition to manual uploading. They are also subject to a *quality check* to exclude sequences presenting anomalous characters or non-conventional amino acids.

The results are saved in a .csv table, which is easier to visualise and download. They are also permanently stored on the server-side and can be accessed at any moment by the user. The output is then available in three formats:.csv, .xls, and .json.

Part of NERVE 2.0 modules were substituted with new packages created ad hoc. *Python-TMHMM*, a protein topology predictor [17] was adapted to fit the new NERVE pipeline. The adhesin predictor SPAAN [18] was completely renewed adopting a different ML architecture and using a new training dataset. The same model was also trained on different datasets to obtain *Virulent*, a module for the prediction of virulent factors. In addition, ML was considered for protein function prediction, performed by DeepFri [19]. Four new components have been added to NERVE 2.0, namely *Loop-Razor*, *Virulent*, *Mouse Immunity* and *Epitope prediction*. A brief description of each component is provided hereafter, while Fig. 1 summarises NERVE 2.0 overall structure.

Users interacting with the web interface or using NERVE 2.0 stand-alone version (see *Availability of data and materials*) can decide to optionally activate only some of these components and modify specific parameters allowing customisable PVCs search.

Subcelloc module

Subcelloc predicts input protein subcellular localizations and aims at finding surface-exposed proteins, which are ideal vaccine targets [18]. For this purpose, this module uses PSORTb 3.0 [20], which shows improved precision and recall with respect to the version 2.0, required for the original NERVE. PSORTb 3.0 predicts the subcellular localization

(See figure on next page.)

Fig. 1 NERVE 2.0 working structure. Bacterial protein sequences are provided as an input FASTA proteome and undergo eight analytical steps: **A** *Subcelloc* predicts protein subcellular localization, **B** *Adhesin* returns the probability of a protein to be an adhesin, **C** *Tmhelices* predicts protein topology, **D** *Loop Razor* rescues membrane proteins reduced to their extracellular fragments, **E** *Autoimmunity and Mouse Immunity* which find respectively matches between the pathogen under analysis and human or mice proteomes **F** *Conservation* which detects conserved proteins between two input bacterial strains, **G** *Virulent* to infer presence of virulence factors and **H** *Annotation to predict protein function*. Then, the *Select* module **I** filters out PVCs, which meet specific requirements. Output results can be downloaded in .json, .csv, or.xlsx format. *Epitope* prediction **J** is performed after the Select module. Created with BioRender.com

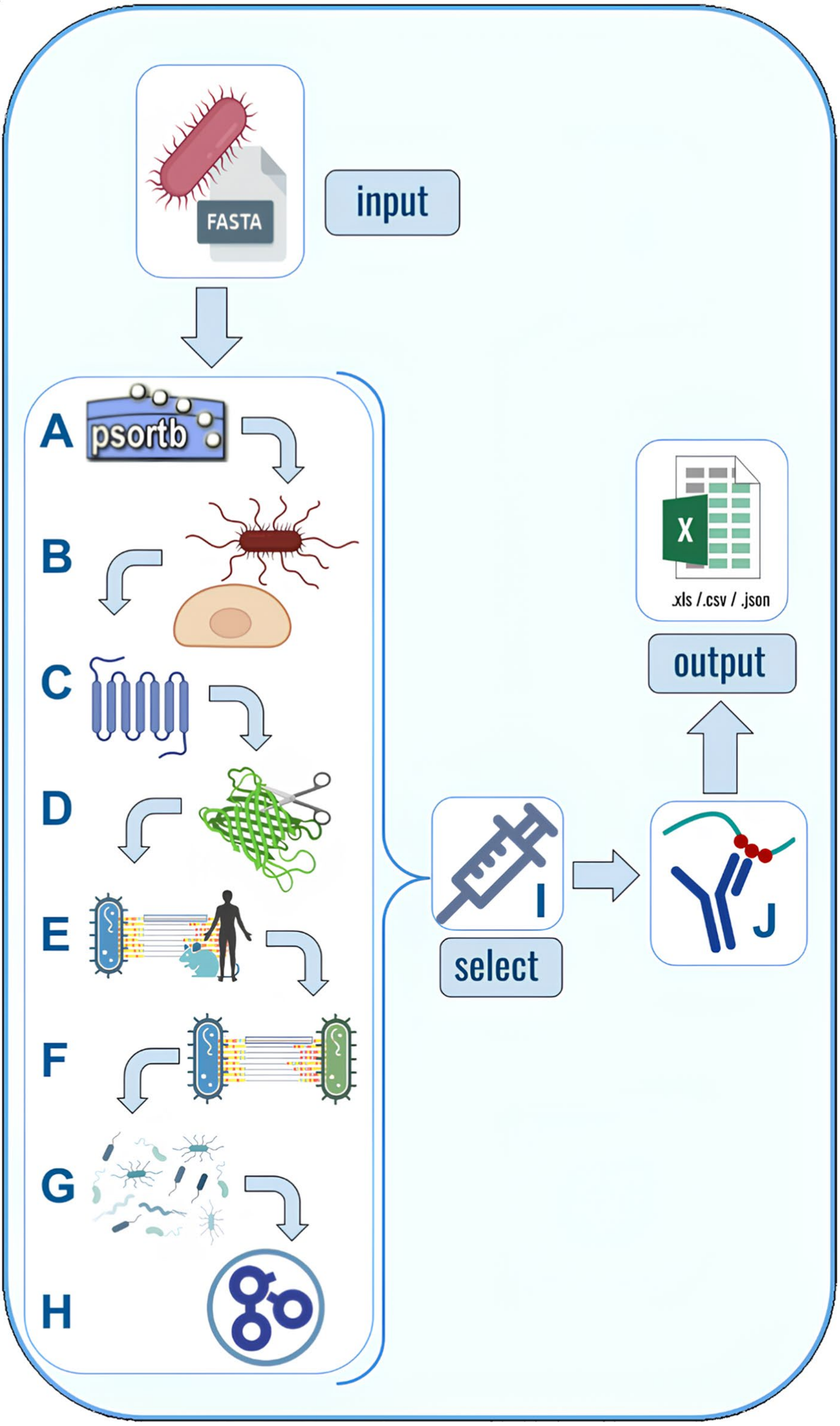


Fig. 1 (See legend on previous page.)

given the bacterial Gram type, which is a classification based on bacterial membrane staining [20]. Predictions equal or above the threshold of 7.5 are considered valid in accordance with the PSORTb 3.0 documentation [20].

ESPAAN, adhesin prediction module

Bacterial adhesins are surface-associated virulence factors that play a crucial role during the first steps of infection, mediating attachment to host cells [21]. Because they are surface bound and required for the infection to occur, they can be readily targeted by the immune system thus representing valid PVCs [21]. A list of known adhesive domains derived from the literature [22] was used to construct a dataset via jackhammer search on reference eubacteria proteomes using the HMMER web server (HMMER 3.3.2) [23]. A set of putative non-adhesin proteins was derived from a search on the SwissProt section (manually reviewed and curated proteins) of Uniprot [16] considering bacterial proteins with non-adhesin related keywords. Redundant sequences (25% identity threshold) were removed from both datasets with CD-hit [24] obtaining 2700 adhesin proteins. A subset of non-adhesin proteins was randomly selected to match the size of the adhesin dataset. Proteins with local similarity above 25% with proteins used to tune the *Select* module (see Sect. "Tuning and benchmarking tests"), as measured with BLASTp [25], have also been removed.

A neural network based on a 10-unit Dense layer was implemented for adhesins identification. The network takes some protein features as input, and it is trained to correctly classify if a protein is an adhesin or not. Cross-entropy loss is used. The features considered are computed with the Python package iFeature [26] and consist of amino acid composition (AAC), dipeptide composition (DPC), composition (CTDC), transition (CTDT), and distribution (CTDD), as similarly done in PathoFact, a tool for the prediction of virulence factors and antimicrobial genes [27]. Since every sequence has a $(20+400+39+39+195=693)$-dimensional feature vector, we performed Principal Component Analysis (PCA) to reduce the dimensionality. We found 400 features to be sufficient to explain the variation observed in the dataset, which was split into 50% training, 25% validation and 25% test set. The training was performed for 120 epochs. To test ESPAAN performance, the main evaluation metrics (shown in Fig. 2 and Table 1) were calculated on its test set.

ESPAAN shows very good performances. Precision and recall both have high values (>0.9), and therefore F1-score is high, too (0.932). Additionally, probabilities of finding false positives and negatives are quite low (0.052 and 0.084 respectively). The 0.932 recorded accuracy demonstrates that *ESPAAN* makes overall correct predictions with few exceptions. In addition, *ESPAAN* was also benchmarked against its predecessor, SPAAN,[1] demonstrating a notable superiority (Table 1).

To better assess *ESPAAN* performances and to avoid overfitting, a threefold cross validation was performed with TensorFlow. As evidenced by data presented in Table 2, *ESPAAN* also shows very high mean validation metrics (>0.9) with very low related standard deviations. This reconfirms the results of the previous demonstration.

[1] https://github.com/kitamura-felipe/adhesin_finder.

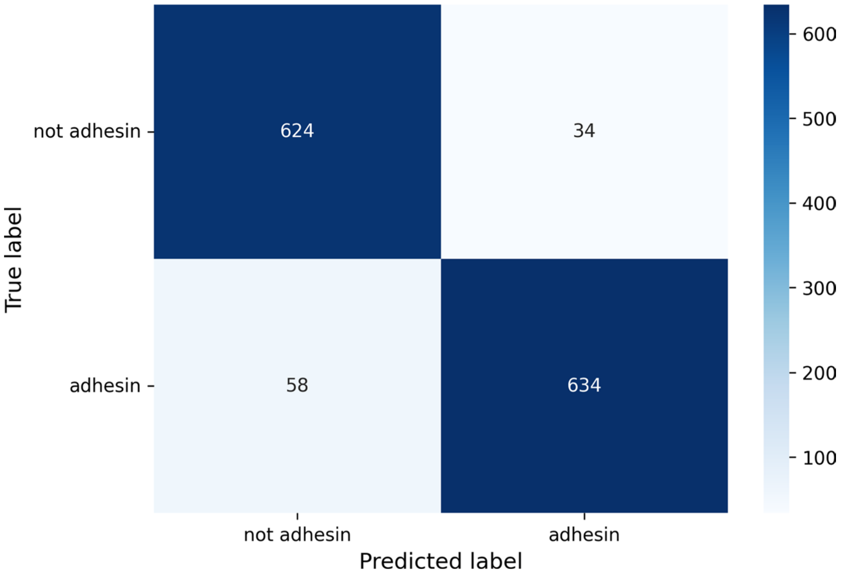


Fig. 2 ESPAAN confusion matrix. A 2 × 2 matrix has been considered for this binary classification problem (adhesin/non-adhesin), setting PAD (probability of being adhesin) = 0.5 as threshold

Table 1 Evaluation metrics of *ESPAAN* and SPAAN, calculated on *ESPAAN* test set using a dedicated python script with TensorFlow

Metric	ESPAAN	SPAAN
Accuracy	0.932	0.669
Sensitivity (recall)	0.916	0.566
Precision	0.949	0.737
F1-score	0.932	0.640
False positive rate (FPR)	0.052	0.220
False negative rate (FNR)	0.084	0.434
True negative rate (TNR)	0.948	0.780
False discovery rate (FDR)	0.051	0.263
Negative predictive value (NPV)	0.915	0.624

The test set consists of 692 positive and 658 negative sequences. 5 sequences from the positive set and 25 from the negative sets were filtered out by SPAAN and therefore not considered to calculate its metrics

See Additional file 1 in *Supplementary Material* and *Availability of data and materials* for an overall comprehension of *ESPAAN* tests data.

TMhelices module

The third step is the protein topology prediction, which consists in finding cytosolic, transmembrane, and extracellular regions. To this aim, we used an ameliorated version of Python-TMHMM, which is based on a hidden-Markov model [17]. We implemented the source code available at https://github.com/dansondergaard/tmhmm.py adapting it to work with thousands of FASTA protein sequences. *TMhelices* predicts, for each protein, the number of transmembrane helices domains (TMDn), as well as"Tmhmm seq", a reduced-alphabet protein sequence containing all different topologies detected, which is

Table 2 Three-fold cross validation of ESPAAN with related obtained metrics

Metric	Value	Std. deviation
Mean validation accuracy	0.929	0.002
Mean validation recall	0.932	0.005
Mean validation precision	0.914	0.002

crucial for *Loop-Razor.* The substitution of HMMTOP [28], the topology predictor used in NERVE 1.0, simplified the installation procedure.

Loop-Razor module

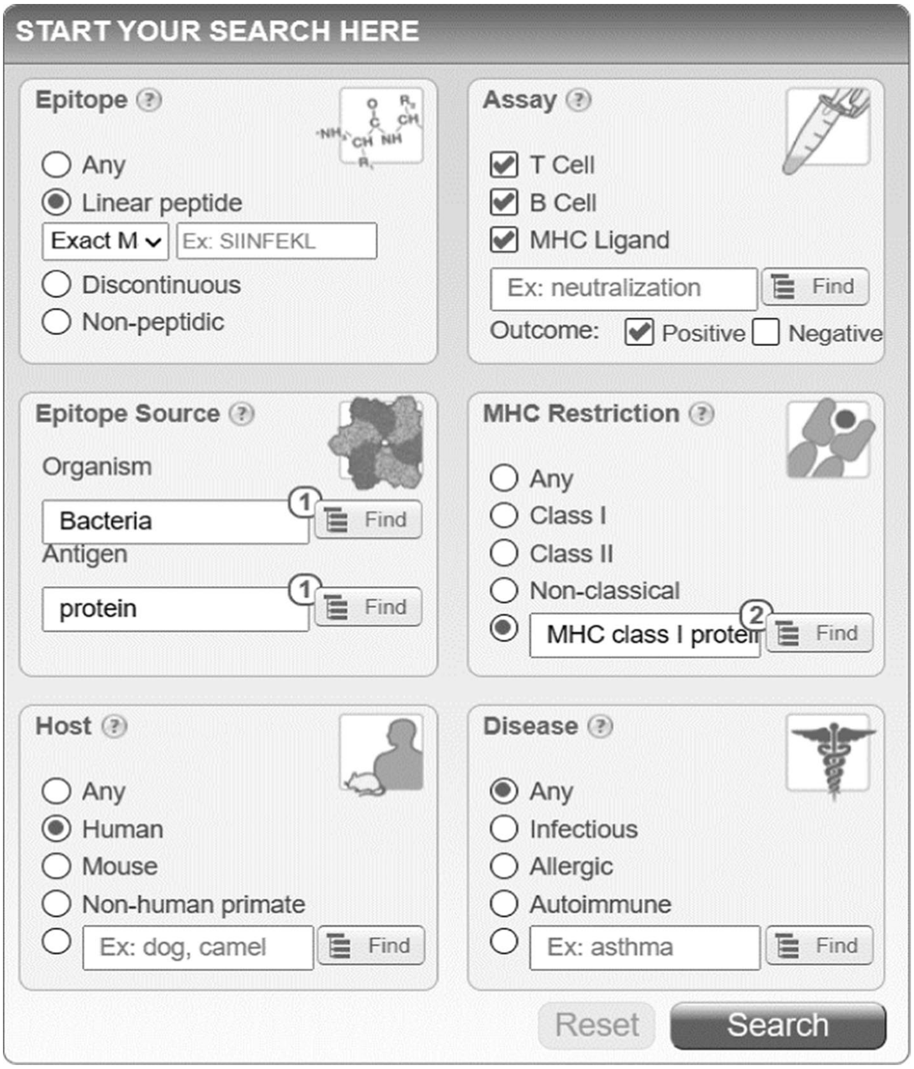
Loop-Razor allows the user to retrieve peptides of PVCs, which have TMDn ≥ 3. It has been introduced because most of the filtration—RV programs, such as NERVE 1.0, discards proteins with TMDn ≥ 3. Such cut-off has been applied so far to avoid impaired expression of the recombinant protein, a very frequent outcome when dealing with membrane proteins, despite several, recently improved protocols [29, 30]. In the pioneering work of Pizza et al. [2], 250 out of the 600 surface proteins of *Neisseria Meningitidis Group B*—endowed with multiple transmembrane domains—were excluded from subsequent characterization steps for unsuccessful cloning. Nevertheless, transmembrane proteins very often turned out to be PVCs because they belong to the surface sub-proteome and hence have more exposed epitopes. To avoid discarding such potentially useful epitopes, as suggested by Olaya-Abril et al. [31], protein fragments that are not embedded in the membrane can be selected for cloning and vaccine testing. To this purpose, when *Loop-Razor* is active, only the outside loops (o-loops) of transmembrane proteins are considered for all proteins with TMDn ≥ 3. With outer membrane proteins (OMPs), the internal loops (i-loops) facing the periplasmic space, are also examined, because they belong to the surface proteome too. In these selected loops, if the longest of a minimum 50 amino acids—continuous sequence (default value, user-modifiable) is detected, it will replace the entire original protein sequence and will then be analysed by the subsequent modules. Considering o-loops, or OMPs-i-loops, is made possible by *TMhelices*, as stated. Reducing the excessive filtration, *Loop-Razor* recovers promising VCs.

Autoimmunity module

This module compares bacterial proteins with human ones to identify similar regions to prevent low immune responses, tolerance issues or autoimmune reactions in vaccine recipients. To achieve this, it retains the two-step structure from the original NERVE:

1. Comparison to the human proteome via BLASTp [25]
2. Analysis of found shared peptides (SPs) to look for Major Histocompatibility Complexes (MHC)-I II human ligands.

MHC class I epitopes stimulate T-lymphocytes cytotoxic activity while MHC class II epitopes promote helper T-cell response which is essential for antibody production [32].



START YOUR SEARCH HERE

Epitope

○ Any
● Linear peptide
Exact M ⌄ Ex: SIINFEKL
○ Discontinuous
○ Non-peptidic

Assay

☑ T Cell
☑ B Cell
☑ MHC Ligand
Ex: neutralization Find
Outcome: ☑ Positive □ Negative

Epitope Source

Organism
Bacteria (1) Find
Antigen
protein (1) Find

MHC Restriction

○ Any
○ Class I
○ Class II
○ Non-classical
● MHC class I protei (2) Find

Host

○ Any
● Human
○ Mouse
○ Non-human primate
○ Ex: dog, camel Find

Disease

● Any
○ Infectious
○ Allergic
○ Autoimmune
○ Ex: asthma Find

Reset Search

Fig. 3 Snapshot taken from the IEDB.org home page. All settings applied to create mhcpep-sapiens are shown. "MHC Restriction" is for MHC class I and II

Presence of such ligands could induce strong autoimmune reactions. The SPs parameters (minimum length, substitution, and mismatch) have been maintained with the same default values and are still modifiable. A new tunable parameter is the e-value, introduced to regulate the number of hits found with BLASTp [25].

The outdated MHCPEP database [33] was replaced with a file containing human MHC ligands retrieved from the IEDB database [34] consisting of 7473 bacterial linear epitopes. In the "Assay" section of the IEDB.org homepage (Fig. 3), all types of experiments with positive outcomes have been considered. No filters have been applied to the "Disease" section, instead. To quantify the similarities between microbial and human proteins, we used the same formula adopted in NERVE 1.0 (see *Select module).*

Mouse immunity module

This facultative step was added to ease possible vaccine manufacturing—pre-clinical studies. Comparison of input protein queries to the mouse proteome is once again performed by BLASTp. Found SPs are scanned to eventually find matches with mouse MHC ligands. To this purpose, a database of 2060 mouse ligands (mhcpep-mouse) was retrieved from IEDB.org. The applied filters on the IEDB webpage are the same used for *Autoimmunity*, except for the "Host", which is Mouse in this case. The SPs parameters are also modifiable in this module.

Conservation module

Conservation has maintained the same features as in NERVE 1.0. It compares a selected bacterial proteome with another one from the same strain/serogroup using BLASTp. The assumption on which this comparison is based, taken from the original article, is: "*the more a PVC is conserved, the more protective the vaccine produced*" [4]. Ranking of PVCs in descending order of conservation is possible with the formula adopted in *Autoimmunity* and *Mouse immunity* modules.

Virulent module

This newly developed optional component predicts the probability of being a virulence factor (PVR). This overcomes a main limitation of NERVE 1.0, which only considers adhesins and adhesin-like proteins as PVCs [3]. Indeed, even though these two latter categories often represent relevant vaccine targets, other protein classes should be examined as well. An example are invasins and toxins, which are both pathogenic and antigenic proteins, therefore promising targets for vaccine development [7].

It is therefore necessary to consider all protein classes that allow microorganisms to: (1) colonise host tissues, have immunomodulatory and/or suppressive properties, or (2) deprive the host of essential nutrients which fall under the definition of virulence factors. Since these factors can lack, or have poor, adhesive properties, *Virulent* solves the gaps left by the search of adhesins alone.

To achieve this purpose, we designed a ML model with the same architecture of *ESPAAN*. Both training and validation datasets were retrieved from the Virulence Factor Database (VFDB) [35] (specifically, the protein core dataset) and from the SwissProt section of UniProt, using keyword search [16]. The following keywords were excluded from the search: *virulent*, *pathogen*, *lethal*, *adhesin*, *adherence*, *biofilm*, *toxin*, *endotoxin*, *exotoxin*, *enterotoxin*, *invasin*, *antiphagocytic*, *motility*, *flagella*, *pilus*, *multidrug*, *subtilisin*, *immunoevasion*, *immunomodulation*, *lipopolysaccharide*, *lipoprotein*, *spore, antibiotic.* After redundancy removal with CD-Hit (25% identity threshold) [24], 1820 virulent factors and 1808 non-virulent factor proteins were left. Then, sequences matching proteins used to tune *Select* have been discarded.

A PCA was applied to perform data reduction, selecting only 400 features. Virulent was trained for 110 epochs and its evaluation metrics are reported in the following Fig. 4 and Table 3.

Differently from *ESPAAN, Virulent* shows reasonable, but not outstanding, performances. Precision and recall have values greater than 0.7. False positive and negative

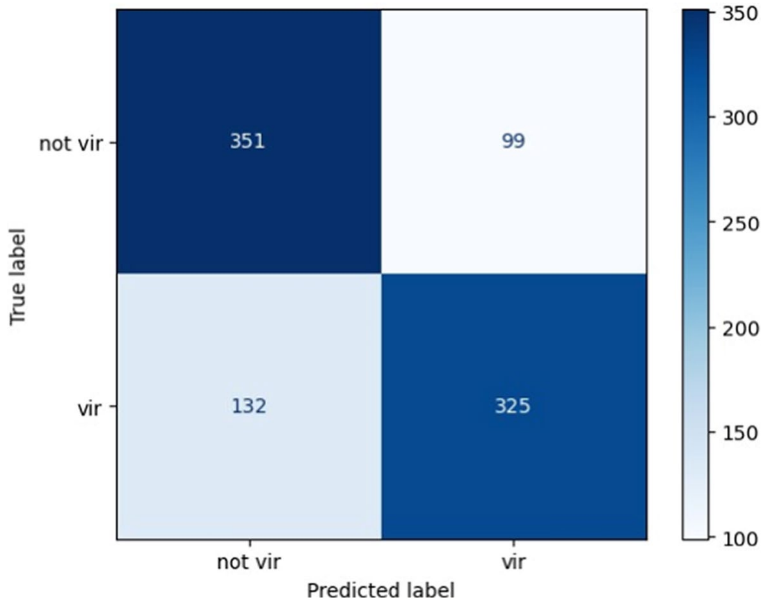


Fig. 4 Virulent confusion matrix. Similarly to *ESPAAN*, here is a 2 × 2 matrix considering PVR (probability of being a virulence factor) = 0.50 as threshold

Table 3 Evaluation metrics of Virulent calculated on its test set, using a dedicated python script with TensorFlow

Metric	Value
Accuracy	0.751
Sensitivity (recall)	0.724
Precision	0.768
F1-score	0.745
False positive rate (FPR)	0.222
False negative rate (FNR)	0.276
True negative rate (TNR)	0.778
False discovery rate (FDR)	0.232
Negative predictive value (NPV)	0.735

rates are not exactly negligible as the *ESPAAN* ones, in particular FDR, which is 0.276. *Virulent* overall accuracy is fairly valid (0.751).

A three-fold cross validation was performed for *Virulent* as well (Table 4). This confirmed the quality of the model.

For a comprehensive analysis of *Virulent* tests, see Additional file 1 in *Supplementary Material* and *Availability of data and materials* sections.

Annotation module

This optional module uses the DeepFRI model to infer protein function [19]. The predictor returns Gene Ontology (GO) terms [36] associated with each protein above the confidence threshold of 0.5, in agreement with DeepFRI documentation. The

Table 4 Three-fold cross validation of Virulent with related obtained metrics

Metric	Value	Std. deviation
Mean validation accuracy	0.767	0.019
Mean validation recall	0.751	0.042
Mean validation precision	0.779	0.012

information obtained is not used by *Select* but can be considered to facilitate the manual screening of VCs.

Select module

The PVCs filtration is accomplished by *Select,* following these rules:

- Being not predicted as cytoplasmic;
- Having a PAD value > *padlimit**;
- Having a TMDn ≤ 2 if *Loop-Razor* is off;
- Having sum of SP with host** database)/protein length < 0.15;
- No match with host ligands database;
- PVR > *virlimit**, if *Virulent* module is active***.

**padlimit* and *virlimit* values are user-modifiable;

**the host is human and/or mouse if the related module is on;

***If *Virulent* is on, proteins with either PVR > *virlimit* or PAD > *padlimit* are selected.

We decided to make the activation of *Select* optional. So, when it is not active, all protein information collected is viewable.

A composite score ranging from 0 to 1 is associated with each antigen as it is meant to prioritise the best PVCs by combining the normalised scores of each predictor. The configuration of *Select* and its parameters is detailed in paragraph *2.12.*

Epitope prediction module

The *Epitope prediction* module identifies epitopes that can potentially bind to MHC-I/II complexes. These molecules, which are codified by the HLAs (Human Leukocyte Antigens) genes, bind and present to T cells the epitopes generated from the intracellular processing of exogenous pathogenic proteins during infections [37]. Given the vast diversity in HLA alleles along with epitopes tridimensional structure and combinations [14, 37], this module considers only linear epitopes and strategically employs the HLA supertypes. This, as defined by Sette and Sidney [38], simplifies the analysis by considering a set of representative HLA alleles to ensure a good coverage of MHC complexes within the world's population. Indeed, the MHC-HLA polymorphism allows to cluster them into sets of molecules that bind largely overlapping peptide repertoires [37]. Specifically, the selected HLA alleles encompass HLA-B*44:03, 07:02, HLA-A24:02, *03:01, *02:01, 01:01, and HLA-DRB115:01, *13:01, *11:01, *08:01, *07:01, *04:01, *03:01, *01:01.

The Python package epitopepredict [39] was used for epitopes prediction considering the frequency distribution of most common HLA alleles derived from the Allele Frequency Net Database [40]. The user, guided by epitope percentile values, determines the threshold for considering the best-ranked proteins, identified by *Select*. Notably, this component also identifies promiscuous epitopes, which binds to multiple alleles simultaneously. This strategy aims to maximise vaccine effectiveness in the single immunised individual and by increasing population coverage. Moreover, the user is free to personalise predicted epitopes length through specific parameters, including the overlap.

Tuning and benchmarking tests

To tune *Select* and benchmark NERVE 2.0 with state-of-the-art web-RV tools, a dataset containing 615 known bacterial protective antigens (BPAs) was derived from the literature [5, 8, 41–43]. Proteins were mapped to Uniprot [16] accession codes and sequence redundancy was removed (90% identity threshold) with CD-Hit [24]. Intracellular proteins are not considered as VCs. Therefore, antigens annotated in Uniprot (version 2023-01) as cytosolic or of unknown localization were excluded. Moreover, since PSORTb 3.0 is a NERVE 2.0 workflow—component, BPAs with local identity *above* 25%, as measured using Blast [25] with proteins from the PSORTb 3.0 training dataset were also excluded to prevent data leakage during these procedures. 153 BPAs were obtained, and the dataset was split into 108 antigens for tuning and 45 for testing and benchmarking. Proteomes and organisms of origin associated with each protein were retrieved from Uniprot. Where no proteome was available, antigens were manually added to the reference proteome of the species. To evaluate *Select* performance during the tuning, the fold enrichment was defined as follows:

$$\text{Fold} - \text{enrichment} = \text{BPAs extracted}/\text{expected bacterial protective antigens (BPAs)}$$

where:

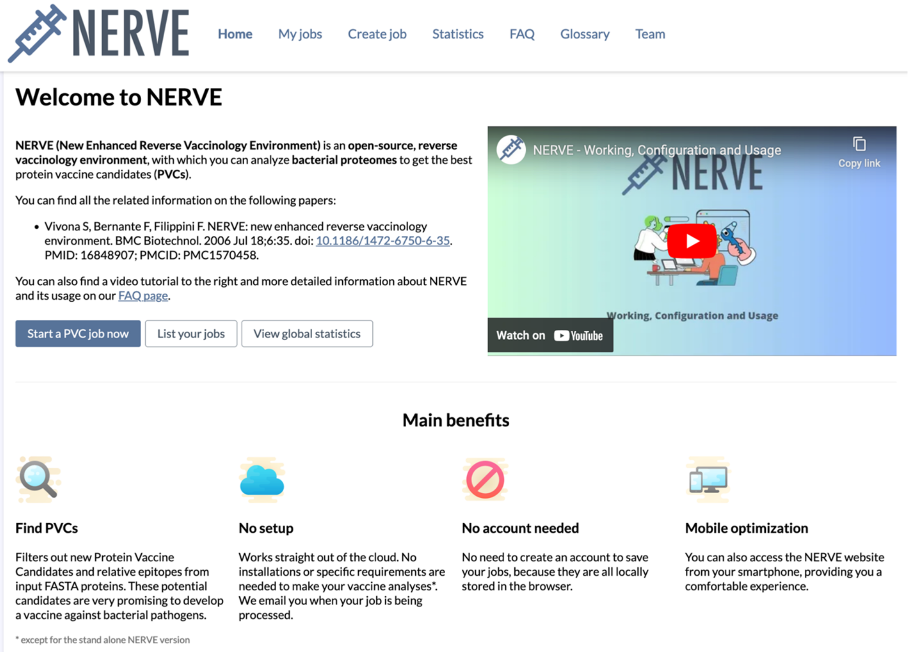
$$\begin{aligned}\text{Expected BPAs} =& (\text{protective vaccine candidates (PVCs)}\\ & *\text{extracted BPAs})/\text{total number}\\ & \text{of proteins submitted}\end{aligned}$$

additionally, the recall was calculated as follows:

$$\text{Recall} = \text{True Positives}/\left(\text{True Positives} + \text{False Negatives}\right)$$

High fold-enrichment values indicate good performances. A hypergeometric test from the python module Scipy (https://scipy.org) was applied to verify statistical significance, as a standard procedure already adopted to test RV programs [5].

It was not possible to obtain a running version of NERVE 1.0 due to its unavailable Perl libraries. NERVE 2.0 was therefore benchmarked against its previous version using the original NERVE 1.0 test dataset, containing 29 BPAs [4]. This dataset was used only for this purpose because it contains several proteins with high similarities with other ones from the PSORTb 3.0 training dataset.



Welcome to NERVE

NERVE (New Enhanced Reverse Vaccinology Environment) is an **open-source, reverse vaccinology environment**, with which you can analyze **bacterial proteomes** to get the best protein vaccine candidates (**PVCs**).

You can find all the related information on the following papers:

- Vivona S, Bernante F, Filippini F. NERVE: new enhanced reverse vaccinology environment. BMC Biotechnol. 2006 Jul 18;6:35. doi: 10.1186/1472-6750-6-35. PMID: 16848907; PMCID: PMC1570458.

You can also find a video tutorial to the right and more detailed information about NERVE and its usage on our FAQ page.

Main benefits

Find PVCs

Filters out new Protein Vaccine Candidates and relative epitopes from input FASTA proteins. These potential candidates are very promising to develop a vaccine against bacterial pathogens.

No setup

Works straight out of the cloud. No installations or specific requirements are needed to make your vaccine analyses*. We email you when your job is being processed.

No account needed

No need to create an account to save your jobs, because they are all locally stored in the browser.

Mobile optimization

You can also access the NERVE website from your smartphone, providing you a comfortable experience.

* except for the stand alone NERVE version

Fig. 5 Snapshot from the NERVE 2.0 homepage, with menu header, main benefits, and the dedicated tutorial video

Website building methods

NERVE Web is an open access software with a web interface and distributed computing mechanism on top of NERVE. The server software is written in Node.js and runs on an Express.js server connected to a Redis database and S3 compliant object storage. Docker is used for container virtualisation. The front-end development is written using the Angular framework.

When a processing job is started, an available cluster node starts to run NERVE with input provided. When a job is finished, the user is notified by email and it can view the PVCs filtered with all their analysed features and download the output file.

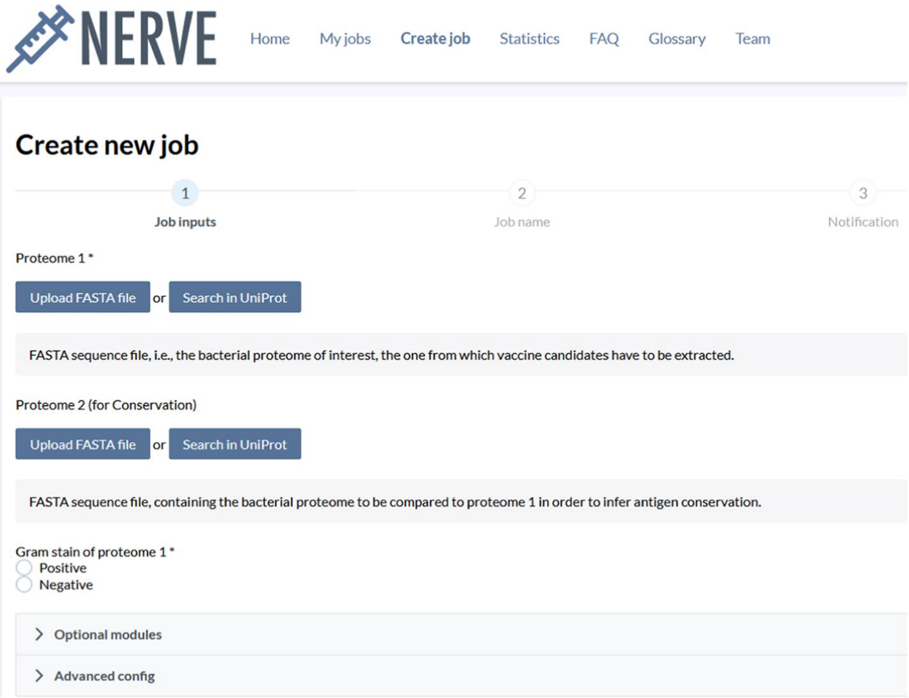
Created jobs and related IDs are stored locally in the browser, so there's no need to create an account.

Results and discussion

In this section, a focus on the website structure, with the site's pages descriptions, is provided. A detailed analysis on NERVE benchmarking tests is reported afterwards.

Website interface

To validate our design choices, both the interface and available functions were tested by volunteers not involved in the project (see *Acknowledgements*) and improved based on their feedback. Hereafter we report on individual sections and the functionality of the web application.



Create new job

Job inputs | Job name | Notification

Proteome 1 *

FASTA sequence file, i.e., the bacterial proteome of interest, the one from which vaccine candidates have to be extracted.

Proteome 2 (for Conservation)

FASTA sequence file, containing the bacterial proteome to be compared to proteome 1 in order to infer antigen conservation.

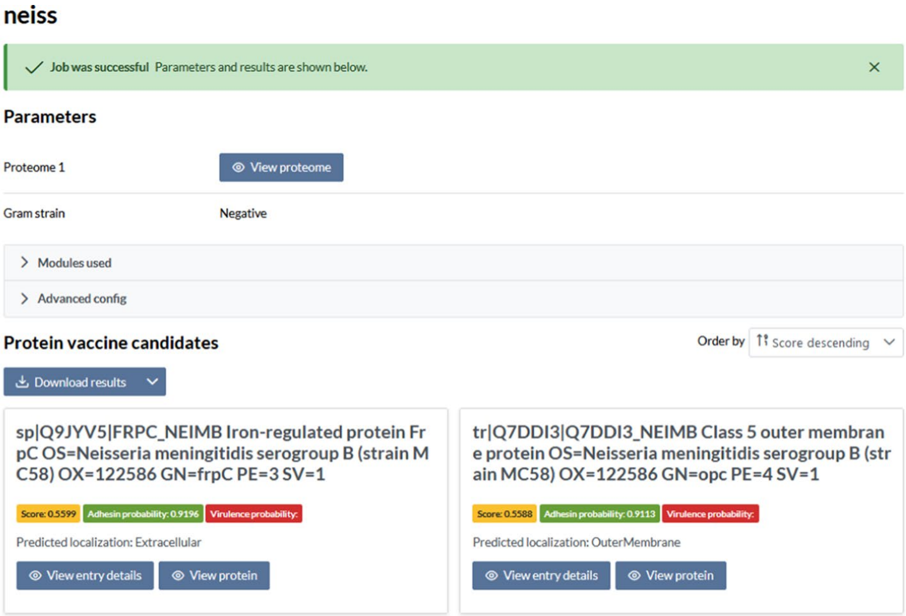
Gram stain of proteome 1 *
○ Positive
○ Negative

Fig. 6 Snapshot from NERVE 2.0 website (Create new job section)

Overview and home page

By clicking on: https://nerve-bio.org/, a user enters the NERVE website homepage (Fig. 5).

In the menu header, there are the links to the seven pages of the site: *Home*, *My jobs*, *Create job*, *Statistics*, *FAQ*, *Glossary* and *Team*. In *My jobs*, all analyses launched by the



neiss

✓ Job was successful Parameters and results are shown below.

Parameters

Proteome 1

Gram strain Negative

Protein vaccine candidates

Order by Score descending

sp|Q9JYV5|FRPC_NEIMB Iron-regulated protein FrpC OS=Neisseria meningitidis serogroup B (strain MC58) OX=122586 GN=frpC PE=3 SV=1

Score: 0.5599 Adhesin probability: 0.9196 Virulence probability:

Predicted localization: Extracellular

tr|Q7DDI3|Q7DDI3_NEIMB Class 5 outer membrane protein OS=Neisseria meningitidis serogroup B (strain MC58) OX=122586 GN=opc PE=4 SV=1

Score: 0.5588 Adhesin probability: 0.9113 Virulence probability:

Predicted localization: OuterMembrane

Fig. 7 Snapshot from NERVE 2.0 website. Results with PVCs filtered from the proteome of *Neisseria meningitidis* group B (MC58)

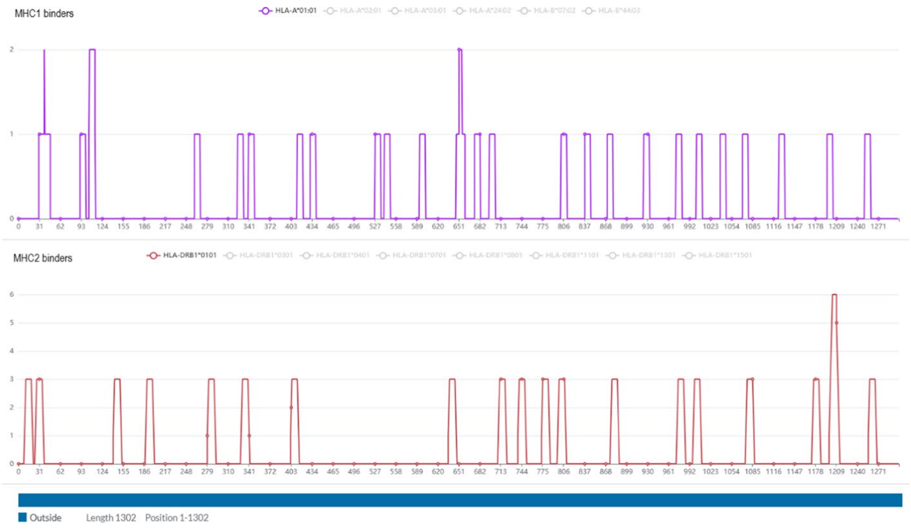


Fig. 8 A detail of *Epitope prediction* results and TMHMM sequence (at the bottom) of UniprotKB: Q9K0K9 extracted from the proteome of *Neisseria meningitidis* group B (MC58)

user are shown, with the possibility to check their status or to look at the results. Clicking on *Create job*, the user can start a new analysis. The *Statistics* section shows the numbers of user-completed, failed, or delayed jobs. In *FAQ*, NERVE features and working are described along with a tutorial. In *Glossary,* a list of bioinformatics related words, with their definitions, are reported to guide the beginners through their first vaccine analyses. Finally, in *Team*, there are the profiles of people who are contributing to the NERVE project.

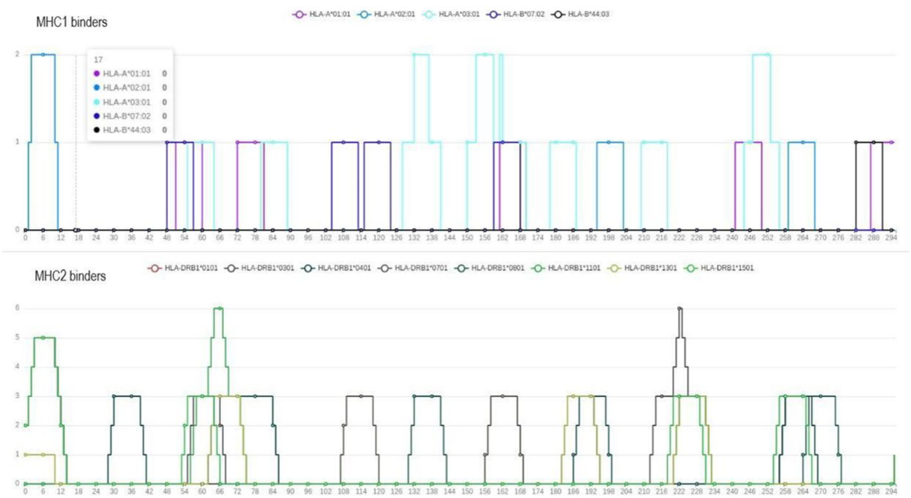


Fig. 9 *Epitope prediction* results for UniprotKB: P17739 from the proteome of *Borrelia burgdorferi (strain ATCC 35210)*. Here are highlighted the MHC-I and II binders for multiple alleles to identify promiscuous linear epitopes

Table 5 Tests and benchmark results of NERVE 2.0: highest fold-enrichment values indicate the best performance

Online RV tools	Modules/methods used			PVCs	Observed BPAs	Recall (%)	Expected BPAs	Fold-enrichment	*p*-value
	Razor	Mouse	Virulent						
NERVE 2.0	No	No	No	796	11	25.00	0.66	16.57	4.21e − 11
	Yes	No	No	828	7	15.91	0.69	10.13	5.29e − 6
	Yes	Yes	No	828	7	15.91	0.69	10.13	5.29e − 6
	Yes	Yes	Yes	4441	11	25.00	3.70	2.97	8.34e − 4
Vaxign2	ML-based			27,227	36	81.82	22.71	1.58	3.10e − 5
	Filter-based			1059	8	18.18	0.88	9.06	2.40e − 6
Vaxijen	ML-based			10,905	31	70.45	9.10	3.41	1.67e − 12

45 known BPAs from 21 proteomes were used and 52,752 proteins derived from the respective proteomes, 36 of which were discarded by NERVE2 after failing quality control analysis. The same dataset was tested on Vaxign2, which failed on performing the computation on 607 proteins, and on Vaxijen. *p*-values have been calculated with a hypergeometric test

Create job

In this section the user uploads its input and sets all the options and parameters for its analysis (Fig. 6).

Results visualisation and download

An example of how job results are visualised is shown in Figs. 7, 8 and 9.

A summary of all settings and components activated is provided at the top of the webpage. In the *Protein vaccine candidates'* section (Fig. 7), the user can see all data collected for each filtered protein by clicking on its related *View entry details*. Here, there is also the Tmhmm seq and the *Epitope prediction* results, with MHC-I and II binders (Fig. 8 and 9).

Benchmarking

To find the best *Select* configuration, we tested three different settings on the tuning dataset described in Sect. "Tuning and benchmarking tests": (1) without *Loop-Razor*, *Mouse immunity* and *Virulent*, (2) with *Loop-Razor* only, (3) with *Loop-Razor* + *Mouse immunity* and (4) with *Loop-Razor* + *Mouse immunity* + *Virulent* combination. For each configuration, the best *padlimit* and *virlimit* thresholds were obtained by performing a four-fold cross validation procedure resulting in 0.5 for both, which were thus set as default values. The immunogenic and TMDn thresholds were not changed from NERVE 1.0 (see Sect. "Select module").

We evaluated each configuration of *Select* on the test dataset by measuring the fold-enrichment and applying the hypergeometric test.

As shown in Table 5, the best performance was (1) *Loop-Razor*, *Mouse immunity* and *Virulent* deactivation, (fold enrichment = 16.57). The activation of *Loop-Razor* and both *Loop-Razor* and *Mouse immunity* produced the same results consisting in a fold-enrichment reduction (10.13), while the additional activation of *Virulent* further decreased the fold-enrichment (2.97).

Performances were benchmarked against existing web-based RV tools on the same test set: Vaxign2 (https://violinet.org/vaxign2) [8] and Vaxijen (http://www.ddg-pharmfac.net/vaxijen/VaxiJen/VaxiJen.html) [9]. Vaxign2 results were retrieved with a customised Python script for web scraping while Vaxijen results were provided by the authors. We used a web scraping approach for Vaxign2 as its command line version was failing the process for some proteomes with cryptic errors, preventing us from running the computation on 607 proteins. Tests were performed on the remaining set. Vaxign2 was tested using either the score obtained with VaxignML (0.9 threshold), a ML model based on eXtreme Gradient Boosting trained to predict BPAs [44] and, with the filtering method (adhesin probability > 0.51, number of transmembrane segments < 2 and extracellular localization as suggested by the website default parameters). Vaxign-ML showed poor performance, predicting a consistent protein amount as BPAs in the dataset (fold-enrichment: 1.58) while the filtering method performed significantly better (fold-enrichment: 9.06). Vaxijen is also based on a ML method and provides a score associated with each protein. Proteins with a score > 0.4 were considered BPAs according to Vaxijen website. Its performance is better than ML-based Vaxign2 but worse than filter-based Vaxign2 (fold-enrichment: 3.41).

NERVE 2.0 also outperformed its previous version, showing a 38.5-fold enrichment (p-value = 1.89e − 41) compared to 8.14 of NERVE 1.0 (p-value = 8.85e − 29) when tested on the NERVE 1.0 test set.

Overall, NERVE 2.0 demonstrated its superior performances compared to both web-based state-of-the-art predictors and to its previous version.

Conclusions

NERVE 2.0 is now available in a new guise, with a simple and clear web interface to be easily and readily usable. So, user-friendliness represents one of the most relevant features of this significant update. Moreover, new components, with related AI models and adjustable parameters guarantee respectively an improved and a customisable computational vaccine analysis, meeting the demands of all kinds of users.

NERVE 2.0 showed better performances compared to its predecessor and to other web-based RV programs (Vaxign2 and Vaxijen). Even if the activation of some components may result in lower fold-enrichment values, this is compensated by evidence that these values are still high, while providing the users with new functionalities and minimising the stringency in selection. The user can narrow the extraction of VCs by setting high stringency. Alternatively, it is possible to choose between more candidates that would have been otherwise discarded, as discussed for the Loop-Razor module, which recovers antigens fragments, or for Virulent, which selects further possible VCs when activated.

Together with the web application, the NERVE 2.0 stand-alone version allows the users to perform high-throughput analyses, not being limited to server requests or bad Internet connection. For its installation, it requires: a Linux-operating system, Docker and a few instructions to follow.

Concerning future perspectives, we will further support NERVE with steady improvements and additions. Thus, progressively optimised AI models for protein analyses will be implemented to provide the users with all the necessary tools to refine their vaccine research.

Abbreviations

RV	Reverse vaccinology
VC	Vaccine candidate
PVC	Protein vaccine candidate
ML	Machine learning
AI	Artificial intelligence
ANN	Artificial neural network
PAD	Probability of being an adhesin
TMDn	Number of transmembrane helices domains
OMP	Outer membrane protein
BPA	Bacterial protective antigen
MHC	Major histocompatibility complex
SP	Shared peptide
PVR	Probability of being a virulence factor
HLA	Human leukocyte antigen
FPR	False positive rate
FNR	False negative rate
TNR	True negative rate
FDR	False discovery rate
NPV	Negative predictive value

Supplementary Information

The online version contains supplementary material available at https://doi.org/10.1186/s12859-024-06004-0.

Additional file 1. NERVE 2.0 supplementary material, with theoretical background of the adopted ML model performance measures

Additional file 2. NERVE 1.0 pipeline, with related caption describing all tools involved and its overall working scheme

Acknowledgements

We thank Alex Bateman, Sandro Vivona, Regina Tavano and Irene Righetto for testing NERVE 2.0, providing us useful feedback.

Author contributions

Conceptualization, A.C., N.G., F.C., M.C. and F.F.; methodology, N.G., F.C., F.B. and F.P.; software, N.G., F.C., F.B., F.P. and A.C..; validation, A.C., F.C. and N.G.; formal analysis, A.C., M.C., F.C. and F.P.; investigation, A.C., F.C., M.C., F.P.; resources, F.C. and N.G..; data curation, A.C., N.G., F.B. and F.C.; writing—original draft preparation, A.C., F.C. and N.G.; writing—review and editing, F.F.; supervision, F.F. All authors have read and agreed to the published version of the manuscript.

Funding

Open access funding provided by Università degli Studi di Padova.

Availability of data and materials

Project name: NERVE 2.0. NERVE home page: https://nerve-bio.org. NERVE stand-alone version: https://github.com/nerve-bio/NERVE. Operating system (web-based version): platform independent. Operating system (stand-alone version): Linux; for other requirements see the GitHub page. Programming language: Python. License: MIT license. Any restrictions to use by non-academics: see https://github.com/nerve-bio/NERVE?tab=MIT-1-ov-file. ESPAAN GitHub page: https://github.com/nicolagulmini/spaan. Virulent GitHub page: https://github.com/nicolagulmini/virulent_factor_classification. ***Tmhelices*** Github page: https://github.com/nicolagulmini/tmhmm.py. Data about testing their performances: https://github.com/nerve-bio/NERVE/tree/main/models_data. List of antigens for test and tuning: https://github.com/nerve-bio/NERVE/blob/main/database/antigens/test_antigens_summary_v2.xlsx. Results of NERVE benchmarking: https://github.com/nerve-bio/NERVE/tree/main/tuning_and_benchmarks.

Declarations

Ethics approval and consent to participate

Not applicable.

Consent for publication

Not applicable.

Competing interests

The authors declare no competing interests.

Received: 14 July 2024 Accepted: 3 December 2024
Published online: 18 December 2024

References

1. Rappuoli R. Reverse vaccinology. Curr Opin Microbiol. 2000;3(5):445–50.
2. Pizza M, Scarlato V, Masignani V, Giuliani MM, Aricò B, Comanducci M, Jennings GT, Baldi L, Bartolini E, Capecchi B, Galeotti CL, Luzzi E, Manetti R, Marchetti E, Mora M, Nuti S, Ratti G, Santini L, Savino S, Scarselli M, Storni E, Zuo P, Broeker M, Hundt E, Knapp B, Blair E, Mason T, Tettelin H, Hood DW, Jeffries AC, Saunders NJ, Granoff DM, Venter JC, Moxon ER, Grandi G, Rappuoli R. Identification of vaccine candidates against serogroup B meningococcus by whole-genome sequencing. Science. 2000;287(5459):1816–20.
3. Cuypers B, Rappuoli R, Brozzi A. A lean reverse vaccinology pipeline with publicly available bioinformatic tools. Methods Mol Biol. 2023;2673:341–56.
4. Vivona S, Bernante F, Filippini F. NERVE: new enhanced reverse vaccinology environment. BMC Biotechnol. 2006;18(6):35.
5. Dalsass M, Brozzi A, Medini D, Rappuoli R. Comparison of open-source reverse vaccinology programs for bacterial vaccine antigen discovery. Front Immunol. 2019;14(10):113.
6. Jaiswal V, Chanumolu SK, Gupta A, Chauhan RS, Rout C. Jenner-predict server: prediction of protein vaccine candidates (PVCs) in bacteria based on host-pathogen interactions. BMC Bioinform. 2013;1(14):211.
7. Rizwan M, Naz A, Ahmad J, Naz K, Obaid A, Parveen T, Ahsan M, Ali A. VacSol: a high throughput in silico pipeline to predict potential therapeutic targets in prokaryotic pathogens using subtractive reverse vaccinology. BMC Bioinform. 2017;18(1):106.
8. Ong E, Cooke MF, Huffman A, Xiang Z, Wong MU, Wang H, Seetharaman M, Valdez N, He Y. Vaxign2: the second generation of the first web-based vaccine design program using reverse vaccinology and machine learning. Nucleic Acids Res. 2021;49(W1):W671–8.
9. Doytchinova IA, Flower DR. VaxiJen: a server for prediction of protective antigens, tumour antigens and subunit vaccines. BMC Bioinform. 2007;5(8):4.
10. Bowman BN, McAdam PR, Vivona S, Zhang JX, Luong T, Belew RK, Sahota H, Guiney D, Valafar F, Fierer J, Woelk CH. Improving reverse vaccinology with a machine learning approach. Vaccine. 2011;29(45):8156–64.
11. Heinson AI, Gunawardana Y, Moesker B, Hume CC, Vataga E, Hall Y, Stylianou E, McShane H, Williams A, Niranjan M, Woelk CH. Enhancing the biological relevance of machine learning classifiers for reverse vaccinology. Int J Mol Sci. 2017;18(2):312.
12. D'Mello A, Ahearn CP, Murphy TF, Tettelin H. ReVac: a reverse vaccinology computational pipeline for prioritization of prokaryotic protein vaccine candidates. BMC Genomics. 2019;20(1):981.
13. Orvis J, Crabtree J, Galens K, Gussman A, Inman JM, Lee E, Nampally S, Riley D, Sundaram JP, Felix V, Whitty B, Mahurkar A, Wortman J, White O, Angiuoli SV. Ergatis: a web interface and scalable software system for bioinformatics workflows. Bioinformatics. 2010;26(12):1488–92.
14. Kaushik R, Kant R, Christodoulides M. Artificial intelligence in accelerating vaccine development—current and future perspectives. Front Bacteriol. 2023;9(2):1258159.
15. TensorFlow: large-scale machine learning on heterogeneous systems. https://www.tensorflow.org/.
16. UniProt Consortium. UniProt: the universal protein knowledgebase in 2023. Nucleic Acids Res. 2023;51(D1):D523–31.
17. Krogh A, Larsson B, von Heijne G, Sonnhammer EL. Predicting transmembrane protein topology with a hidden Markov model: application to complete genomes. J Mol Biol. 2001;305(3):567–80.
18. Sachdeva G, Kumar K, Jain P, Ramachandran S. SPAAN: a software program for prediction of adhesins and adhesin-like proteins using neural networks. Bioinformatics. 2005;21(4):483–91.
19. Gligorijević V, Renfrew PD, Kosciolek T, Leman JK, Berenberg D, Vatanen T, Chandler C, Taylor BC, Fisk IM, Vlamakis H, Xavier RJ, Knight R, Cho K, Bonneau R. Structure-based protein function prediction using graph convolutional networks. Nat Commun. 2021;12(1):3168.
20. Yu NY, Wagner JR, Laird MR, Melli G, Rey S, Lo R, Dao P, Sahinalp SC, Ester M, Foster LJ, Brinkman FS. PSORTb 3.0: improved protein subcellular localization prediction with refined localization subcategories and predictive capabilities for all prokaryotes. Bioinformatics. 2010;26(13):1608–15.
21. Raynes JM, Young PG, Proft T, Williamson DA, Baker EN, Moreland NJ. Protein adhesins as vaccine antigens for Group A Streptococcus. Pathog Dis. 2018;76(2):fty016.
22. Monzon V, Lafita A, Bateman A. Discovery of fibrillar adhesins across bacterial species. BMC Genomics. 2021;22(1):550.
23. Potter SC, Luciani A, Eddy SR, Park Y, Lopez R, Finn RD. HMMER web server: 2018 update. Nucleic Acids Res. 2018;46(W1):W200–4.
24. Li W, Godzik A. Cd-hit: a fast program for clustering and comparing large sets of protein or nucleotide sequences. Bioinformatics. 2006;22(13):1658–9.
25. Altschul SF, Madden TL, Schäffer AA, Zhang J, Zhang Z, Miller W, Lipman DJ. Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. Nucleic Acids Res. 1997;25(17):3389–402.
26. Chen Z, Zhao P, Li F, Leier A, Marquez-Lago TT, Wang Y, Webb GI, Smith AI, Daly RJ, Chou KC, Song J. iFeature: a Python package and web server for features extraction and selection from protein and peptide sequences. Bioinformatics. 2018;34(14):2499–502.
27. de Nies L, Lopes S, Busi SB, Galata V, Heintz-Buschart A, Laczny CC, May P, Wilmes P. PathoFact: a pipeline for the prediction of virulence factors and antimicrobial resistance genes in metagenomic data. Microbiome. 2021;9(1):49.
28. Tusnády GE, Simon I. The HMMTOP transmembrane topology prediction server. Bioinformatics. 2001;17(9):849–50.
29. Claassens NJ, Siliakus MF, Spaans SK, Creutzburg SCA, Nijsse B, Schaap PJ, Quax TEF, van der Oost J. Improving heterologous membrane protein production in *Escherichia coli* by combining transcriptional tuning and codon usage algorithms. PLoS ONE. 2017;12(9): e0184355.
30. Pedro AQ, Queiroz JA, Passarinha LA. Smoothing membrane protein structure determination by initial upstream stage improvements. Appl Microbiol Biotechnol. 2019;103(14):5483–500.
31. Olaya-Abril A, Jiménez-Munguía I, Gómez-Gascón L, Obando I, Rodríguez-Ortega MJ. Identification of potential new protein vaccine candidates through pan-surfomic analysis of pneumococcal clinical isolates from adults. PLoS ONE. 2013;8(7): e70365.

32. He Y. Bacterial whole-genome determination and applications. In: Molecular medical microbiology, 2nd edn, 2014. p. 357–68.
33. Brusic V, Rudy G, Harrison LC. MHCPEP, a database of MHC-binding peptides: update 1997. Nucleic Acids Res. 1998;26(1):368–71.
34. Vita R, Mahajan S, Overton JA, Dhanda SK, Martini S, Cantrell JR, Wheeler DK, Sette A, Peters B. The immune epitope database (IEDB): 2018 update. Nucleic Acids Res. 2019;47(D1):D339–43.
35. Liu B, Zheng D, Jin Q, Chen L, Yang J. VFDB 2019: a comparative pathogenomic platform with an interactive web interface. Nucleic Acids Res. 2019;47(D1):D687–92.
36. The Gene Ontology Consortium. The gene ontology resource: 20 years and still going strong. Nucleic Acids Res. 2019;47(D1):D330–8.
37. Sidney J, Peters B, Frahm N, et al. HLA class I supertypes: a revised and updated classification. BMC Immunol. 2008. https://doi.org/10.1186/1471-2172-9-1.
38. Sette A, Sidney J. HLA supertypes and supermotifs: a functional perspective on HLA polymorphism. Curr Opin Immunol. 1998;10(4):478–82.
39. Farrell D. Epitopepredict: a tool for integrated MHC binding prediction. GigaByte. 2021.
40. Gonzalez-Galarza FF, McCabe A, Santos EJ, Jones J, Takeshita LY, Ortega-Rivera ND, Del Cid-Pavon GM, Ramsbottom K, Ghattaoraya GS, Alfirevic A, Middleton D, Jones AR. Allele frequency net database (AFND) 2020 update: gold-standard data classification, open access genotype data and new query tools. Nucleic Acid Res. 2020;48:D783–8.
41. Ricci AD, Brunner M, Ramoa D, Carmona SJ, Nielsen M, Agüero F. APRANK: computational prioritization of antigenic proteins and peptides from complete pathogen proteomes. Front Immunol. 2021;15(12): 702552.
42. He Y, Racz R, Sayers S, Lin Y, Todd T, Hur J, Li X, Patel M, Zhao B, Chung M, Ostrow J, Sylora A, Dungarani P, Ulysse G, Kochhar K, Vidri B, Strait K, Jourdian GW, Xiang Z. Updates on the web-based VIOLIN vaccine database and analysis system. Nucleic Acids Res. 2014;42(Database issue):D1124–32.
43. Ansari HR, Flower DR, Raghava GP. AntigenDB: an immunoinformatics database of pathogen antigens. Nucleic Acids Res. 2010;38(Database issue):D847–53.
44. Ong E, Wang H, Wong MU, Seetharaman M, Valdez N, He Y. Vaxign-ML: supervised machine learning reverse vaccinology model for improved prediction of bacterial protective antigens. Bioinformatics. 2020;36(10):3185–91.

Publisher's Note