Proteomics
Proteomics and Systems Biology

Data-Independent Acquisition Mass Spectrometry and Machine Learning Algorithms in the early-stage diagnosis of pancreatic cancer from clinical plasma samples

Journal:	*PROTEOMICS*
Manuscript ID	Draft
Wiley - Manuscript type:	Research Article
Date Submitted by the Author:	n/a
Complete List of Authors:	Nouri Nigjeh, Benjamin; Vertex Pharmaceuticals Incorporated,
Keywords from curated list:	data mining < Bioinformatics, data processing and analysis < Bioinformatics, body fluids < Biomedicine, cancer < Biomedicine, mass spectrometry - LC-MS/MS < Technology
Keywords:	

SCHOLARONE™
Manuscripts

Data-Independent Acquisition Mass Spectrometry and Machine Learning Algorithms in the early-stage diagnosis of pancreatic cancer from clinical plasma samples

For Peer Review

Benjamin Nouri Nigjeh[1*]

[1*] Department of Medicine, University of Washington at Seattle, WA 98195, USA

Address all correspondence to: nigjeh@uw.edu or benjamin.nigjeh@gmail.com

Abstract:

Pancreatic cancer is the most lethal tumor entity, and it will soon become the second most frequent cause of cancer-related death in the Western world. Lack of reliable plasma biomarkers for early-stage diagnosis necessitates the global quantification of circulating proteins. Previously, we have shown that data-independent acquisition mass spectrometry is capable of quantification of 1898 circulating proteins from depleted clinical plasma samples from 19 early-stage pancreatic cancer samples versus 19 healthy controls (n=38). In this study, neural network algorithms are optimized and trained on a training dataset (n=28) with k-fold and leave-one-out cross validations. The performance of the trained model on prediction accuracy of a hold-out test dataset (n=10) was examined with per-sample diagnosis using a plethora of hyperparameters, i.e., activation functions, optimizers, and learning rates. The neural network algorithm was benchmarked with several classic machine learning algorithms in sensitivity versus specificity of diagnosis based on AUC values derived from ROC graphs. Successful training of the neural network algorithm from the training dataset was monitored based on a leave-one-out cross validation function. Trained neural network algorithm alongside logistic regression, multinomial naive bayes, and random forest algorithms showed accurate prediction of the hold-out test plasma samples.

For Peer Review

Keywords: Mass Spectrometry, Proteomics, Data Independent Acquisition, Machine Learning, Neural Network Algorithms, Deep Learning

Introduction

Pancreatic cancer is the most lethal tumor entity, and it will soon become the second most frequent cause of cancer-related death in the Western world[1]. The most common subtype of pancreatic cancer is pancreatic ductal adenocarcinoma (PDAC) which constitutes 90 to 95% of all the cases [1, 2, 3]. Surgy still remains the most viable therapy option, but it is most effective (based on the 5 years survival rate studies) only when performed at early-stages of the disease[1]. Diagnosis of pancreatic cancer at early-stages of disease is particularly challenging, due to the lack of reliable biomarkers and position of pancreas in body. Embedded position of pancreas, and its small size complicates high-resolution imaging techniques and tissue biopsies for early-stage diagnosis of pancreatic cancer. In contrast, plasma collection is a non-invasive procedure and plasma contains valuable systemic information underlying all the body organs and tissues[4]. This property makes plasma a good biological matrix candidate for diagnosis purposes. However, there is a lack of reliable plasma biomarkers to develop targeted blood diagnosis assays using ELISA and targeted proteomics[1]. Previously, it was shown that simultaneous quantification of more than 300 differential proteins may result in highly sensitive and specific diagnosis of pancreatic cancer according to AUC values[5]. This result is encouraging for the development of comprehensive global proteomic quantification technologies for the diagnosis of pancreatic cancer at early stages of disease from plasma samples.

Data-independent acquisition mass spectrometry is a technology capable of global quantification of proteins using consecutive narrow precursor ion ion-isolation windows with monitoring the ion-fragment signal intensities. This technology is supported with the advent of high resolution and high frequency mass spectrometry instruments capable of generation of comprehensive proteomic datasets containing quantitative information underlying expression of thousands of proteins from complex biological mixtures. Recently, development of gas-phase ion mobility separations has enabled multi-dimensional diaPASEF technology, based on the parallel ion accumulation and serial fragmentation combined to attain even deeper proteomic analysis[6]. Data independent acquisition technology has shown promises in elucidation of sub-proteome analysis such as glycosylation[7], and in particular for oncology purposes[8]. These developments make data independent acquisition mass spectrometry a sound technology for the comprehensive and in-depth quantification of plasma proteins. In a previous study, our group optimized a data independent acquisition mass spectrometry approach capable of quantifying 1898 plasma proteins in a wide analysis dynamic range with high level of reproducibility, which was monitored by a spiked in proten[9]. The technique was used to quantify 19 early-stage pancreatic ductal adenocarcinoma samples against 19 healthy controls.

For Peer Review

Learning from global proteomic datasets generated by data independent acquisition mass spectrometry requires training machine learning algorithms [10, 11, 12]. A well-trained algorithm will be beneficial in binary and categorical classification of pancreatic cancer samples analyzed by mass spectrometry. However, training machine learning algorithms from proteomic datasets is particularly challenging as it contains high dimensional data structure with high level of biological noises (environmental or genetic), and undesired complexities originated from sample storage and processing[10]. This is echoed by limited number of early-stage pancreatic cancer samples for training purposes that increases the risk of data memorization and overfitting rather than generalization.

Due to the complexity of the proteomic samples, any linear and nonlinear relationship among quantified proteins is plausible. In this study, a fully connected neural network algorithm was optimized to learn from the training proteomic dataset. The architecture of neural network and its hyperparameters are optimized to allow the learning from the training dataset, and to avoid memorization using evaluation functions against k-fold cross validation strategies. The performance of the algorithm was evaluated on a hold-out test dataset on a per-sample analysis basis with multiple replicates. Finally, the optimal neural network algorithm was benchmarked with several classic machine learning models.

For Peer Review

Experimental procedure

Hardware and Software: Machine learning algorithms were developed on an NVIDIA GeForce RTX 3050 GPU and an Intel Core i5 CPU with 8 GB of RAM. The operating system was Windows 11. The computations on NVIDIA GPU were parallelized with the installation of CUDA/cuDNN. Coding was done on Python version 3.9.7 running on PyCharm IDE version 2022.1. Sequential fully connected neural networks were built using keras API, a high-level frontend module integrated on tensorflow library version 2.8. Data visualizations were done either using matplotlib library or on GraphPad Prism version 5.

Plasma Proteomic Dataset: Clinical plasma samples were collected from 19 early-stage pancreatic ductal adenocarcinoma patients and 19 healthy control objects. Plasma sample collection details, preparation of samples, and LC/MS setup for proteomic analysis is described at **supplementary experimental procedure**. Plasma spectral library containing fragmentation pattern from pooled plasma proteins were generated from a previously published spectral library based on spectral matching from two-dimensional separation of depleted pooled plasma samples[9]. Data independent acquisition LC/MS runs from disease versus control objects were deposited in proteomeXchange repository with ID number PXD037127. The raw DIA data was analyzed using the Skyline software. The transitions were limited to the peptides within 410 to 900 m/z, and only top five b or y ions with *m/z* value greater than 200 were selected for data mining. The peptide profiles were explored through the highest library dot products within the 10 minutes of retention time window matched from the spectral library. The signals from the transitions were summed to provide the quantification for the corresponding peptide. Similarly, the signals from peptides for each protein were summed to obtain the quantification at protein level. The quantification protein table included quantitative information from 1898 plasma protein and 10871 unique peptides, which is available from the repository. The protein intensities from each protein were normalized to the most abundant sample, so that protein intensities were confined between 0 and 1. The proteomic dataset split into balanced training (n=28) and test (n=10) datasets.

For Peer Review

Results and Discussion:

Training neural networks from proteomic datasets can be challenging due to the high level of background noise and limited number of informative features in the high dimensional proteomic datasets[13]. The background noise might be originated from genetic or environmental variations[14], or they are an artifact generated during sample collection, storage, and processing. To evaluate the performance of neural network algorithms in training from high dimensional proteomic datasets with limited informative features, a contrived noise training dataset of 784 features was generated with only one informative feature with increasing signal to noise ratio. The training dataset contained 1000 control samples versus 1000 experiment samples, with 20% of samples were shuffled and excised for evaluation. The trained algorithm was tested on the binary classification of 200 hold-out test dataset. A two-layer structure algorithm was regularized by an earlystopping callback function to avoid overfitting, and to provide sufficient generalization and prediction. The prediction power of the trained algorithm in binary classification of the test dataset with increasing signal to noise ratio of a stand-alone informative feature is shown in **Figure 1**. Accordingly, when there is no informative feature in the dataset, the algorithm behaves as a random classifier, but by increasing the signal to noise ratio from 2 to 4, the prediction accuracy of the algorithm increases from 68% to 90%. This result emphasizes the power of fully connected neural networks when they are trained from minimally informative datasets and when the algorithm is regularized automatically by an earlystopping callback function.

For Peer Review

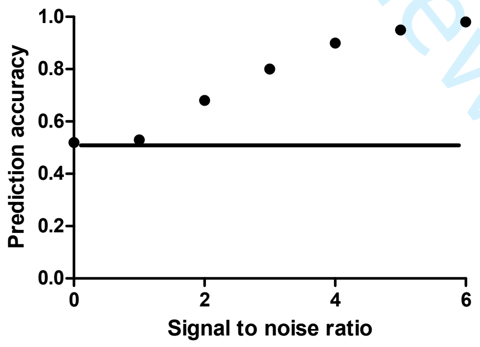

Figure 1. Prediction accuracy of the optimized fully connected neural network algorithm from a test dataset (n=200) under earlystopping callback regularization from the training noise dataset (n=2000) with a single informative feature with increasing signal to noise ratio.

In our previous study, we developed a data-independent acquisition mass spectrometry technique capable of quantification of 1898 circulating proteins from 19 early-stage pancreatic ductal adenocarcinoma (PDAC) samples versus 19 healthy controls. The method was verified by the quantification reproducibility of a spiked in protein throughout the quantification of clinical samples (i.e., Invertase 2)[9]. The fragmentation signal of tryptic peptides from the spiked protein quantified in the cohort of clinical samples didn't change significantly, highlighting the reproducibility and reliability of the quantification[9]. The circulating protein levels were measured simply by the sum of the area under curve from their corresponding quantified tryptic peptides. **Figure 2** shows the volcano graph of the 1898 quantified proteins from cancer samples versus healthy controls. Accordingly, there are 58 differentially quantified proteins identified that are differentially expressed in the pancreatic cancer patient samples with fold changes more than 2, and p-values less than 0.05. Gene Ontology based analysis of these differentially expressed proteins showed that these proteins belong mainly to immune response as well as metabolic disorder biological processes (**Figure 2**, right panel). Though these changes are highly relevant to pancreatic cancer pathology, but they are in common with a plethora of other diseases causing inflammatory responses and metabolic disorders such as chronic pancreatitis. This leads to the fact that there is a need for a holistic learning from the differential proteomic changes for an accurate diagnosis of cancer.

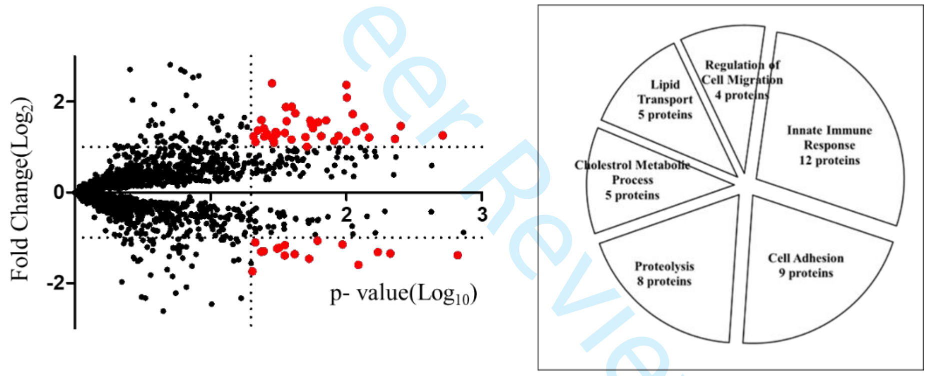

Figure 2. Volcano graph from 19 early-stage PDAC plasma samples versus 19 healthy controls. Proteins with fold changes more than 2, and p-values less than 0.05 are shown in red (left panel). The gene ontology-based annotation of biological processes that differentially expressed proteins belong to (right panel, n=58).

Holistic learning from quantitative proteomic datasets requires training machine learning algorithms. Beside the high level of background noise present in the proteomic datasets, the limited number of training samples (in particular with high dimensional proteomic datasets) may result in data memorization due to the presence of excessive number of trainable parameters [15, 16]. In this study, a hold-out test dataset containing 5 cancer samples and 5 healthy controls (**test dataset** in supplementary information) were excised for the performance evaluation of the trained algorithm. The remaining training dataset contained 14 cancer samples and 14 healthy controls (**training dataset** in supplementary information). Due to the limited number of training samples, k-fold cross validation, and leave-one-out cross validation functions were used to design the optimal

For Peer Review

architecture of the neural network, and to assess the learning process throughout increasing learning cycles, respectively.

The architecture of the neural network was optimized using keras tuner function and k-fold cross validation of 4 within the training dataset[17]. As presented in **code S1**, the performance of the algorithm with two hidden neural network layers in each iteration is evaluated with the ¼ of the dataset. This approach provided the optimal numbers of the nodes per layer (150 nodes for first layer, and 200 for second layer) for optimal training from the training dataset. The activation function on the nodes was rectified learning unit ("relu"), and the backpropagation optimizer was "adam", a stochastic gradient descent method that is based on adaptive estimation of first order and second-order moments with learning step of 0.001. Next, the training of the algorithm by increasing number of training epochs was evaluated against the leave-one-out cross validation (see **code S2**). As shown in **Figure 3**, the training loss of the neural network algorithm throughout training is compared against the evaluation loss. The evaluation loss trend indicates that the algorithm continues learning from the training dataset up to epochs c.a. 50, whereas increased number of training cycles caused overfitting, data memorization, and loss of the generalization. The sharp drop in training loss by increasing epoch number is also an indicative for data memorization. In addition, **Figure 3** implies that the training plasma dataset contains informative features that can be useful for training and binary classification of the cancer samples from healthy controls.

For Peer Review

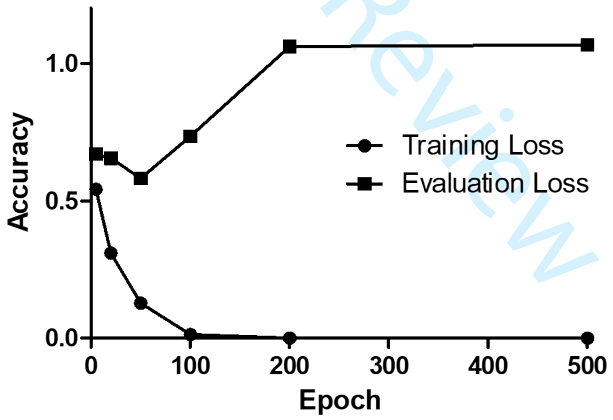

Figure 3. Training loss versus evaluation loss by increasing number of training epoch estimated from leave-one-out cross validation when algorithm was trained with the training dataset

The performance of the trained neural network algorithm with optimal number of nodes per layers was evaluated against the prediction accuracy of the hold-out test dataset. **Figure 4** shows the per-sample illustration of the prediction from 5 algorithm training replicates. The algorithm shows the top performance in classification of the test samples at epoch number of 50, corresponding to the minima in evaluation loss curve in **Figure 3**. Training with less epoch number showed the insufficient learning from the training dataset while training with larger epoch numbers resulted in the loss of generalization, and overfitting, highlighted with the algorithm performance good only for a subset of samples.

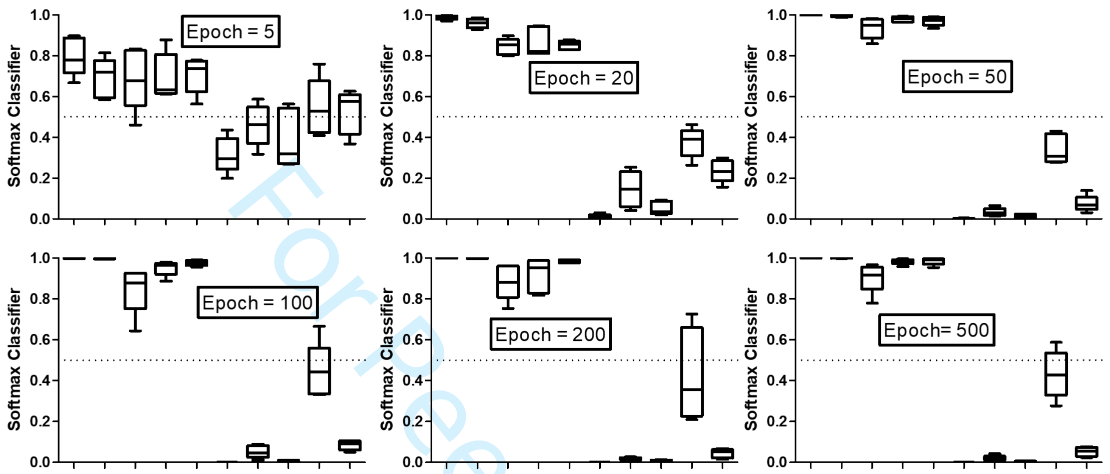

Figure 4. The impact of increasing training epoch cycles on prediction accuracy of plasma test dataset with 5 algorithm training replicates (First five samples are healthy controls test samples and next five are cancer plasma samples).

The learning power of neural networks from complex biological datasets, mainly relies on the presence of a nonlinear activation function for the nodes residing on the hidden layers, and the optimizer performing the backpropagation optimization. **Figure 5** shows the impact of using different nonlinear activation functions when the number of nodes, layers, and training epoch is constant. Under the given conditions, “relu” activation function has a better classification performance when compared with other activation functions, i.e., “elu”, “selu”, and “Tanh”.

For Peer Review

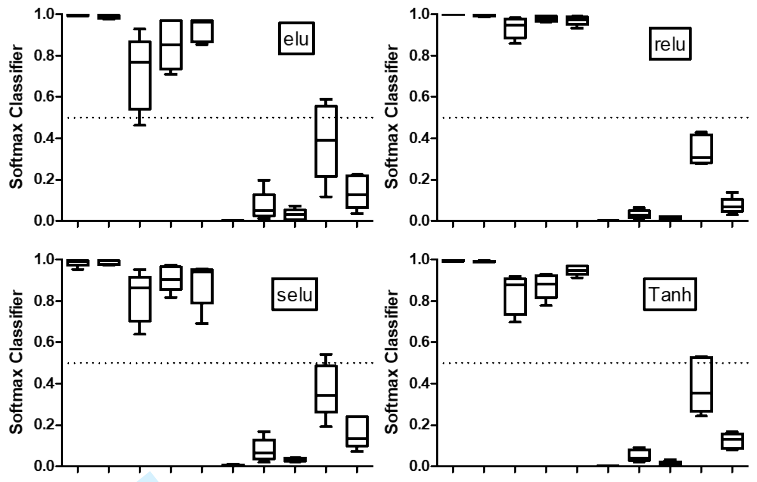

Figure 5. The impact of using different nonlinear activation functions on the prediction accuracies of the trained neural network algorithm in classification of the hold-out test dataset.

Similarly, the performance of the algorithm with different optimizer functions and learning rates were assessed in the light of prediction accuracy from the test dataset. Using the "adam" optimizer with increasing learning rates from 0.05 to 0.001, in **Figure 6**, showed improvement in classification quality of the trained algorithm, while in shorter learning rates the algorithm performance dropped significantly. The adaptive momentum optimizer with long learning rates clearly jumps over informative topologies in the dataset, while the very short learning rates trap the learner algorithm in the local minima. In compare with other optimizers 'adam' with learning rate of 0.001 showed the best performance, while optimizers such as 'nadam', 'adamx', and 'RMSProp' were also able to predict the sample labels correctly. This comparative experiment enables the choice of suitable optimizers for proteomics datasets.

For Peer Review

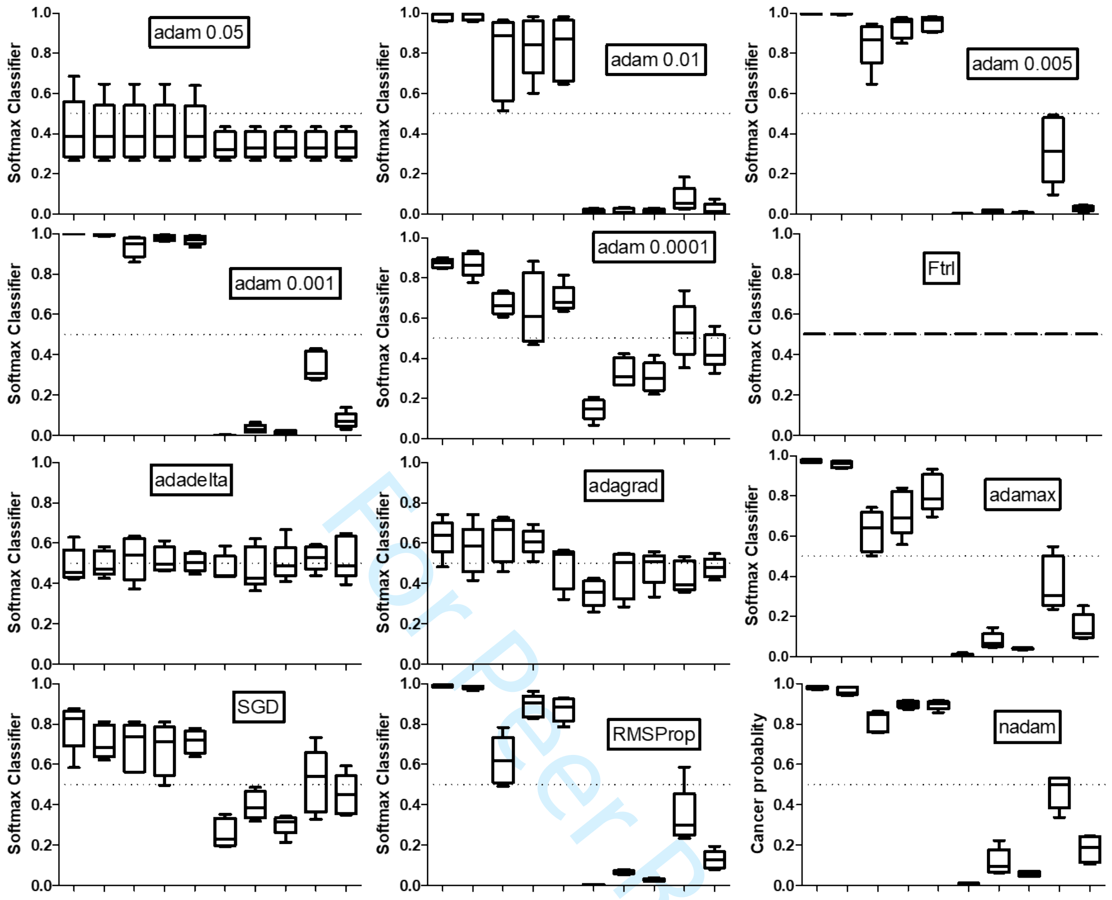

Figure 6. Impact of using different optimizers and learning rates on training from plasma dataset with a neural network algorithm performance evaluated on prediction accuracies of plasma test dataset.

For Peer Review

The performance of the trained neural network algorithm was benchmarked with other classic machine learning models[17] plotted in ROC graphs and shown in **Figure 7** to indicate the sensitivity and specificity of the trained algorithms in prediction of the test dataset. The trained neural network was able to correctly classify the entire test dataset (AUC =1). Here, a model training class was written to receive the parameters of each benchmarked classic machine learning algorithm and generate the corresponding ROC graph as the output (**code S3**). In this study, three Bayesian algorithms (Bernoulli, Multinomial, and Gaussian) were compared with logistic regression, k-nearest neighbors, decision tree, random forest, and support-vector machine algorithms. From these classic machine learning algorithms, three of them were able to assign entire test dataset correctly (logistic regression, random forest, and multinomial Bayesian). Previously, a comparative study also highlighted an identical performance with logistic regression and support-vector machine algorithms in compare with neural networks[18], while decision trees showed the worst performance[18]. The comparisons here also showed a similar trend between neural network and the benchmarked classic algorithms. This result is of particular interest as by assembling different algorithms, it is possible to obtain even better predictive performances from multiple models.

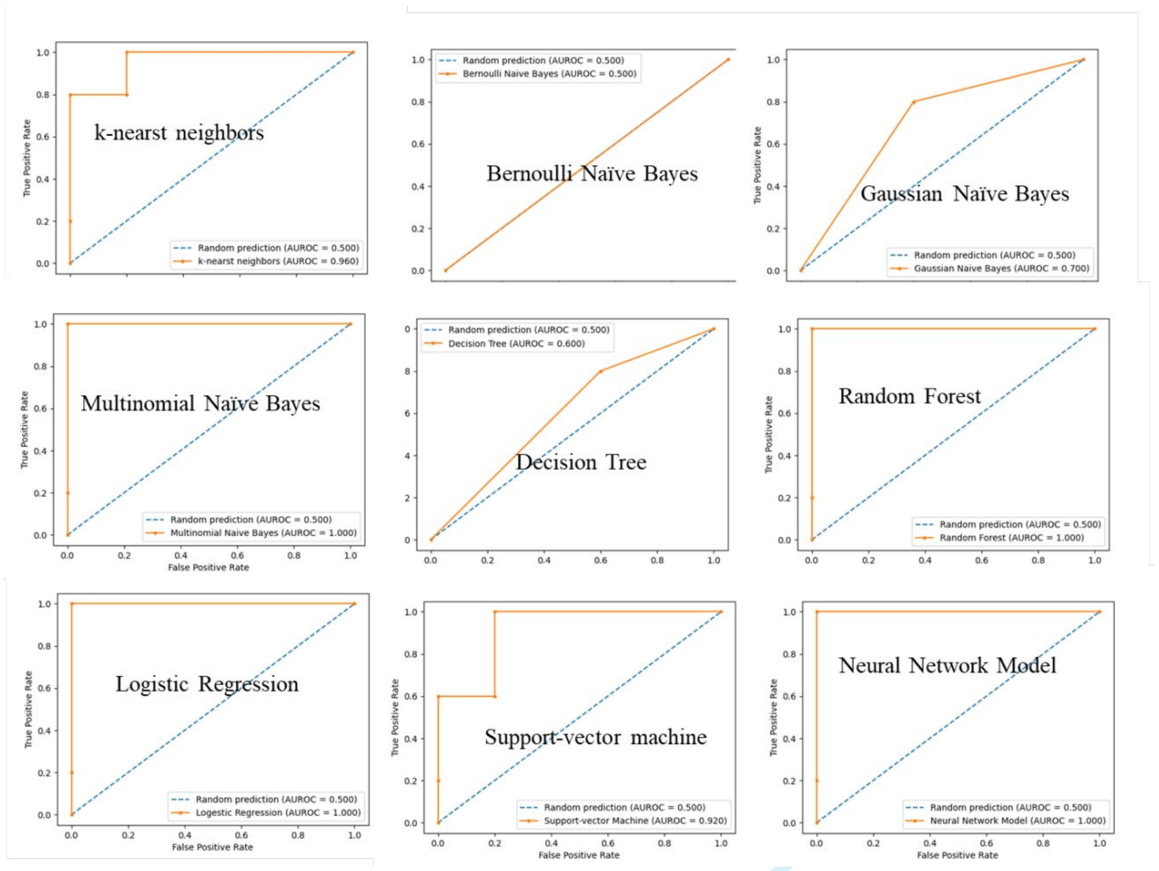

Figure 7. ROC plots of eight different classic machine learning algorithms, and the optimized neural network model trained on the plasma training dataset and tested on the test plasma protein dataset

Conclusions:

Neural network algorithms are able to learn from high dimensional contrived noise datasets with as little as one informative feature, implicating for training from proteomic datasets with high biological background noise levels. The architecture of the neural network algorithm was optimized against evaluation with k-fold cross validation function to achieve the optimal number of nodes per hidden layers and trainable parameters for a given dataset. The learning process of the neural network algorithm was monitored against leave-one-out cross validation function with increasing number of epoch for training loss versus evaluation loss. It was evident that the neural network algorithm is learning from the proteomic dataset with an optimal epoch number of 50. The algorithm performance in prediction accuracy of a hold-out test dataset was evaluated with 5 algorithm training replicates and was compared with algorithms trained with higher and lower numbers of epochs. Trained algorithm showed success in correct prediction of the hold-out test plasma dataset. The impact of other model hyperparameters (activation function, optimizer, and learning rate) were assessed by prediction accuracy of the test dataset. Finally, neural network algorithm was benchmarked with several classic machine learning algorithms in sensitivity versus specificity estimation of diagnosis based on the area under curve from ROC graphs. Our results showed successful classification of hold-out test plasma samples with logistic regression, random forest, and multinomial naïve bayes algorithm models, alongside neural networks.

For Peer Review

SUPPORTING INFORMATION:

Supplementary experimental procedure describes processes for clinical sample collection, preparation, and LC/MS instrumentation. Raw mass spectrometry files from data independent acquisition mass spectrometry, and the global protein quantification table is available from proteomeXchange repository with ID number PXD037127. **Code S1** in supplementary information presents the layer node number optimization using keras tuner library optimized against k-fold cross validation. **Code S2** presents evaluation of the algorithm performance against leave-one-out cross validation. **Code S3** includes training of several machine learning algorithms alongside optimal neural network evaluated based on area under the curve from ROC graphs of test dataset

For Peer Review

Acknowledgment:

The author appreciates the useful feedback and support from the colleagues Dr. Sheng Pan, from university of Texas, and Dr. Ru Chen, from Baylor college of medicine. A generous grant from Amazon Web Services (AWS) enabled the use of their cloud-based machine learning tools.

References:

(1) Al-Shaheri F. N.; Alhamdani M. S. S.; Bauer A. S.; Giese N.; Büchler M. W.; Hackert T.; Hoheisel J. D. Blood biomarkers for differential diagnosis and early detection of pancreatic cancer. Cancer Treat Rev. 2021, 96, 102193.

(2) Silwal-Pandit L.; Stalberg S. M.; Johansson H. J.; Mermelekas G.; Lothe I. M. B.; Skrede M. L.; Dalsgaard A. M.; Nebdal D. J. H.; Helland A.; Lingjarde O. C.; Labori K. J.; Skalhegg B. S.; Lehtio J.; Kure E. H. Proteome Analysis of Pancreatic Tumors Implicates Extracellular Matrix in Patient Outcome. Cancer Res. Commun. 2022, 2, 434-446.

(3) Kamisawa T.; Wood L. D.; Itoi T.; Takaori K. Pancreatic Cancer. The Lancet. 2016, 388, 73-85.

(4) Chen R.; Yi E. C.; Donohoe S.; Pan S.; Eng J.; Cooke K.; Crispin D. A.; Lane Z.; Goodlett D. R.; Bronner M. P.; Aebersold R.; Brentnall T. A. Pancreatic Cancer Proteome: The Proteins That Underlie Invasion, Metastasis, and Immunologic Escape. Gastroentrology. 2005, 129, 1187-1197.

(5) Mustafa S.; Pan L.; Marzoq A.; Fawaz M.; Sander L.; Rückert F.; Schrenk A.; Hartl C.; Uhler R.; Yildirim A.; Strobel O.; Hackert T.; Giese N.; Büchler M.; Hoheisel J.; Saeed Alhamdani M. Comparison of the tumor cell secretome and patient sera for an accurate serum-based diagnosis of pancreatic ductal adenocarcinoma. Oncotarget 2017, 8(7), 11963–11976.

(6) Meier F.; Brunner A. D.; Frank M.; Ha A.; Bludau I.; Voytik E.; Kaspar-Schoenefeld S.; Lubeck M.; Raether O.; Bache N.; Aebersold R.; Collins B. C.; Rost H. L.; Mann M. diaPASEF: parallel accumulation-serial fragmentation combined with data-independent acquisition. Nat Methods. 2020, 17, 1229-1236.

(7) Ye Z.; Vakhrushev S. Y. The Role of Data-Independent Acquisition for Glycoproteomics. MCP. 2021, 20, 100042.

(8) Krasny L.; Huang P. H. Data-independent acquisition mass spectrometry (DIA-MS) for proteomic applications in oncology. Mol Omics. 2021, 17, 29-42.

(9) Nouri Nigjeh E.; Chen R.; Brand R. E.; Petersen G. M.; Chari S. T.; von Haller P. D.; Eng J. K.; Feng Z.; Yan Q.; Brentnall T. A.; Pan S. Quantitative proteomics based on optimized data-independent acquisition in plasma analysis. J. Proteome Res. 2017, 16, 665–676.

For Peer Review

(10) Kelchtermans P.; Bittremieux W.; De Grave K.; Degroeve S.; Ramon J.; Laukens K.; Valkenborg D.; Barsnes H.; Martens L. Machine learning applications in proteomics research: how the past can boost the future. Proteomics. 2014, 14, 353-66.

(11) Meyer J. G. Deep learning neural network tools for proteomics. Cell Rep. 2021, 1, 100003.

(12) Kim H.; Kim Y.; Han B.; Jang J. Y.; Kim Y. Clinically Applicable Deep Learning Algorithm Using Quantitative Proteomic Data. J. Proteome Res. 2019, 18, 3195–3202.

(13) Grapov D.; Fahrmann J.; Wanichthanarak K.; Khoomrung S. Rise of Deep Learning for Genomic, Proteomic, and Metabolomic Data Integration in Precision Medicine. OMICS. 2018, 22, 630-636.

(14) Enroth S.; Johansson Å.; Enroth S.; Gyllensten U. Strong effects of genetic and lifestyle factors on biomarker variation and use of personalized cutoffs. Nat. Commun. 2014, 5, 4684.

(15) Francois Chollet. Deep Learning with Python; MANNING. 2021. 2nd Edition.

(16) LeCun Y.; Bengio Y.; Hinton G. Deep learning. Nature. 2015, 521, 436-44.

(17) Aurelien Geron. Hands-On Machine Learning with Scikit-Learn and Tensorflow: Concepts, Tools, and Techniques to Build Intelligent Systems; O'REILY. 2017. 3rd Edition.

(18) Lancashire L. J.; Lemetre C.; Ball G. R. An introduction to artificial neural networks in bioinformatics--application to complex microarray and mass spectrometry datasets in cancer studies. Brief Bioinform. 2009, 10(3), 315-329.

For Peer Review

Supplementary Information:

Data-Independent Acquisition Mass Spectrometry and Machine Learning Algorithms in the early-stage diagnosis of pancreatic cancer from clinical plasma samples

For Peer Review

Benjamin Nouri Nigjeh[1*]

[1*] Department of Medicine, University of Washington at Seattle, WA 98195, USA

Address all correspondence to: nigjeh@uw.edu or benjamin.nigjeh@gmail.com

Table of content:

For Peer Review

Supplementary experimental procedure:

Plasma Samples: This study was approved by the Institutional Review Board at the University of Washington (Seattle, WA). Plasma samples were collected from healthy subjects (NL) and early-stage pancreatic ductal adenocarcinoma (PDAC). The diagnosis of disease was made histologically in the case of pancreatic cancer patients. The PDAC patients with early-stage disease were operable, representing a mixture of localized pancreatic cancer (stages 1 and 2). The cancer patients involved in this study did not receive any treatment prior to blood draw. The blood samples were processed using similar protocols within 4 hours after specimen collection. The plasma samples were collected into purple top tubes (Becton Dickinson, Franklin Lakes, NJ, USA) with EDTA, the potassium salt, as an anticoagulant. The blood was centrifuged at 330 × g for 20 minutes. The resultant plasma samples were aliquoted and stored in −80 °C until used.

For Peer Review

Plasma Sample preparation: Invertase 2 standard was added to crude plasma upfront to an initial plasma concentration of 25 µg/mL. For depletion of abundant plasma proteins, a top 12 depletion spin columns (Thermo Fisher Scientific) was used. Six µL of plasma sample was directly loaded onto the buffer at the top of slurry, and the column was incubated with an end-over-end incubator for 1 hour at room temperature. The end of the column was removed, and the sample was eluted with a centrifuge at 100 ×g for 2 minutes. The depleted mixture was collected, and buffer exchanged to 50 mM ammonium bicarbonate using the 5 kDa spin filter, VIVASPIN 500 (Sigma Aldrich), and the final volume was adjusted to 100 µL.

The samples were then de-glycosylated by adding 0.5 µL of PNGase F (New England Biolabs, Ipswich, MA, USA), and incubated for 6 hours at 37 °C. The samples were reduced by adding 10 mM TCEP and incubation at 50 °C for 1 hour and alkylated by adding 25 mM iodoacetamide (IDA) and incubation for 30 minutes at room temperature in the dark. After adjusting the pH to 7.5–8.5 with sodium hydroxide solution, the depleted plasma was digested with MS grade trypsin (Thermo Fisher Scientific, Waltham, MA, USA) at 1:30 enzyme to protein ratio, in a two-step fashion to improve digestion efficiency. In the first step, half of the trypsin was added and the mixture was incubated for 2 hours at 37 °C with vortexing every 30 minutes, then the remaining trypsin was added and the mixture was incubated for additional 16 hours at 37 °C. The digestion was terminated by adding 0.1% formic acid (V/V). Digested samples were dried completely and re-suspended in 0.1% formic acid. One µg of each of the fractions was loaded for LC MS/MS analysis using data independent acquisition (DIA) method. The disposable 12 depletion column was chosen for DIA analysis of individual plasma samples to avoid inter-sample cross contaminations and achieve greater coverage of low abundant proteins.

LC-MS/MS set-up for data-independent analyses: The LC separation set up consisted of a trapping column and an analytical column connected back-to-back to increase the loading speed. The trapping column was a 360 µm and 100 µm outer and inner diameters self-packed integraFit column (Scientific Instrument Services, Ringoes, NJ, USA). The trapping column was packed to a length of about 3 cm with ProntoSIL 200Å (pore size) 5 µm (particle size)-C18 AQ (Mac-Mod, Chadds Ford, PA, USA). The analytical column was 25 cm long with 360 µm and 75 µm outer and inner diameters fused silica packed with ProntoSIL 120 Å -5µm-C18 AQ (Mac-Mod). A column tip was prepared by pulling the column with Laser Fiber Puller P-2000 (Sutter Instruments, Novato, CA, USA). Separation was done with a nanoACQUITY UPLC system from Waters. Buffer A and buffer B were water and acetonitrile with 0.1 % formic acid respectively. 1 µg of sample was loaded on the trapping column with 2% B at 2 µL/min flow rate for 10 minutes. Peptides were then resolved with a 90-minute gradient of 5 to 30% B followed by flushing at 80% B for 10 minutes and column equilibration with 2% B for 20 minutes. The analytical flow rate was 0.3 µL/min and entire data acquisition lasted for 120 minutes. DIA quantification was performed on a QE+ mass spectrometer (ThermoFisher Scientific) with the resolution of 17,500 at 200 m/z. AGC target was set at 1e6 with 55 msec max injection time. Optimal isolation windows were 7 m/z, and NCE was 28, the acquisition window covered a mass range from 410 to 900 m/z through 70 consecutive isolation windows.

For Peer Review

Code S1: Hyperparameter optimization

```
import numpy as np
import pandas as pd
import tensorflow as tf
from sklearn.model_selection import KFold
from keras_tuner.tuners import RandomSearch

file_path = '///training_dataset.csv'
df = pd.read_csv(file_path)
Y = df.pop("target")
X = np.array(df)
y = np.array(Y)

def build_model(hp):
    model = tf.keras.models.Sequential()
    model.add(tf.keras.layers.Dense(hp.Int("input_units_1", min_value =100,
max_value =200, step =5), activation=tf.nn.relu))
    model.add(tf.keras.layers.Dense(hp.Int("input_units_2", min_value =100,
max_value =300, step =5), activation=tf.nn.relu))
    model.add(tf.keras.layers.Dense(2, activation=tf.nn.softmax))
    model.compile(optimizer='adam',
                  loss='sparse_categorical_crossentropy',
                  metrics='accuracy')
    return model

tuner = RandomSearch(
    build_model,
    objective= "val_accuracy",
    max_trials= 500,
    executions_per_trial=1,
    directory= "log_dir1"
    )


kf = KFold(n_splits=4, shuffle=True, random_state=100)
kf.get_n_splits(X)

for train_index, evaluation_index in kf.split(X):
    X_train, X_evaluation = X[train_index], X[evaluation_index]
    y_train, y_evaluation = y[train_index], y[evaluation_index]
    tuner.search(x=X_train,
                 y=y_train,
                 batch_size=62,
                 validation_data=(X_evaluation, y_evaluation))
```

Code S2: Training and Evaluation loss

```python
import numpy as np
from sklearn.model_selection import LeaveOneOut
import pandas as pd
import tensorflow as tf

file_path = '///training_dataset.csv'
df = pd.read_csv(file_path)
Y = df.pop("target")
X = np.array(df)
y = np.array(Y)

loo = LeaveOneOut()
loo.get_n_splits(X)

evaluation_loss = []
train_loss = []


for train_index, evaluation_index in loo.split(X):
    X_train, X_evaluation = X[train_index], X[evaluation_index]
    y_train, y_evaluation = y[train_index], y[evaluation_index]
    model = tf.keras.models.Sequential()
    model.add(tf.keras.layers.Dense(150, activation=tf.nn.relu))
    model.add(tf.keras.layers.Dense(200, activation=tf.nn.relu))
    model.add(tf.keras.layers.Dense(2, activation=tf.nn.softmax))
    model.compile(optimizer='adam',
                  loss='sparse_categorical_crossentropy',
                  metrics='accuracy')
    model.fit(X_train, y_train, epochs=1000)
    evaluation_loss_t, ___ = model.evaluate(X_evaluation, y_evaluation)
    train_loss_t, ___ = model.evaluate(X_train, y_train)
    evaluation_loss.append(evaluation_loss_t)
    train_loss.append(train_loss_t)

train_loss = np.reshape(train_loss, (-1,1))
evaluation_loss = np.reshape(evaluation_loss, (-1,1))

result = np.concatenate((evaluation_loss, train_loss), axis = 1)
np.savetxt('///result.csv', result, delimiter=',')
```

Code S3: Machine Learning algorithms

```
from sklearn.naive_bayes import GaussianNB
from sklearn.naive_bayes import BernoulliNB
from sklearn.metrics import roc_curve, roc_auc_score
from sklearn.neighbors import KNeighborsClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.naive_bayes import MultinomialNB
from sklearn.tree import DecisionTreeClassifier
from sklearn import svm
from sklearn.ensemble import RandomForestClassifier
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import tensorflow as tf

# import training dataset
file_path = '///training_dataset.csv'
X = pd.read_csv(file_path)
Y_train = X.pop("target")
X_train = X.copy()
X_train = np.array(X_train)
Y_train = np.array(Y_train)

# import test dataset
file_path = '///test_dataset.csv'
X = pd.read_csv(file_path)
Y_test = X.pop("target")
X_test = X.copy()
X_test = np.array(X_test)
Y_test = np.array(Y_test)

#Define a prediction class from any machine learning algorithm
def predict(model, X_train, Y_train, X_test):
    clf = model.fit(X_train, Y_train)
    a = clf.predict_proba(X_test)
    return a[:, 1]

# KNN algorithm
KN_probs = predict(KNeighborsClassifier(n_neighbors=15, metric='l2',
weights='distance'),
                 X_train, Y_train, X_test)
KN_auc = roc_auc_score(Y_test, KN_probs)

# LR algorithm
LR_probs = predict(LogisticRegression(penalty='l2',
multi_class='multinomial', solver='lbfgs'),
                X_train, Y_train, X_test)
LR_auc = roc_auc_score(Y_test, LR_probs)

#BNB algorithm
BNB_probs = predict(BernoulliNB(),
                X_train, Y_train, X_test)
BNB_auc = roc_auc_score(Y_test, BNB_probs)

# NMB algorithm
```

```
MNB_probs = predict(MultinomialNB(alpha=1.0, class_prior=None,
fit_prior=True),
                X_train, Y_train, X_test)
MNB_auc = roc_auc_score(Y_test, MNB_probs)

# DT algorithm
DT_probs = predict(DecisionTreeClassifier(),
                X_train, Y_train, X_test)
DT_auc = roc_auc_score(Y_test, DT_probs)

# RF algorithm
RF_probs = predict(RandomForestClassifier(max_features= 5, max_depth=2,
n_estimators= 500, random_state=0),
                X_train, Y_train, X_test)
RF_auc = roc_auc_score(Y_test, RF_probs)

# GNB algorithm
GNB_probs = predict(GaussianNB(), X_train, Y_train, X_test)
GNB_auc = roc_auc_score(Y_test, GNB_probs)

# SVM algorithm
SVM_probs = predict(svm.SVC(probability=True), X_train, Y_train, X_test)
SVM_auc = roc_auc_score(Y_test, SVM_probs)

# Neural network algorithm with ROC visualization
model = tf.keras.models.Sequential()
model.add(tf.keras.layers.Dense(150, activation=tf.nn.relu))
model.add(tf.keras.layers.Dense(200, activation=tf.nn.relu))
model.add(tf.keras.layers.Dense(2, activation=tf.nn.softmax))
model.compile(optimizer= 'adam',
              loss='sparse_categorical_crossentropy',
              metrics='accuracy')
model.fit(X_train, Y_train, epochs=50)
a = model.predict(X_test)
model_probs = a[:, 1]
model_auc = roc_auc_score(Y_test, model_probs)

r_probs = [0 for _ in range(len(Y_test))]
r_auc = roc_auc_score(Y_test, r_probs)

r_fpr, r_tpr, _ = roc_curve(Y_test, r_probs)
nb_fpr, nb_tpr, _ = roc_curve(Y_test, model_probs)
plt.plot(r_fpr, r_tpr, linestyle='--', label='Random prediction (AUROC =
%0.3f)' % r_auc)
plt.plot(nb_fpr, nb_tpr, marker='.', label='Neural Network (AUROC = %0.3f)' %
model_auc)
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')

plt.legend()
plt.show()
```