POLITECNICO DI TORINO

MASTER'S DEGREE IN COMPUTER ENGINEERING

MASTER'S DEGREE THESIS

PACKET CONTENT PREDICTION IN A TELEPATHOLOGY SESSION

Supervisors

Prof. ALESSIO SACCO

Prof. GUIDO MARCHETTO

Candidate

NICY MALANDA SENDO

November 2020

Abstract

Telepathology refers to practicing pathology from a distance. Telecommunications technology is used for facilitating the transmission of pathology image data between two distant locations for diagnosis, research and education purposes. In order to perform telepathology, a pathologist must choose the video images that need to be analyzed and then render a diagnosis. The use of television microscopy, which preceded telepathology, didn't require a pathologist to have a virtual or physical hands-on involvement in choosing the microscopic fields-of-view to analyze and diagnose. Today, With the wide applications of prediction, especially in the telemedicine field, the research of prediction algorithm and theory has made a great progress. However, these applications have consistently need to face packet losses and a recovery mechanism is required to achieve accetable delays. The goal of this thesis is to propose an approach that uses a machine learning (ML) method, Hidden Markov Model (HMM), to predict the future packet content from the generated network traffic during a telepathology session. Then, the predicted packet will be used to remedy packet loss and maintain an acceptable latency of data transfers. The thesis work is also focused on the comparison between HMM and others ML tecniques such as Support Machine Vectors (SVM), Autoregressive Integrated Moving Average (ARIMA), Generative Adversarial Network (GAN), etc. Additionally, this approach makes use of the software LiveMicro [1], which is a new edge computing-based telepathology system that enables live histological image processing and real-time remote control of the microscope. During the experimental analysis, we train and test the models with two datasets that represent two different scenarios observed in different telepathology sessions. In particular, HMM achieves the highest average classification accuracy of 87.28% and 82.84%, for the *scenario1* and *scenario2* respectively. These findings indicate that the proposed method, HMM, is the best algorithm analyzed for this study.

Keywords: machine learning, hidden markov model, prediction, packet content, telepathology.

Acknowledgements

I definitely would like to thank to Prof. Guido Marchetto for the opportunity to work on this theme, it's a very convincing theme and it's from the area that I like the most, the IT department. I would also like to express my very profound gratitude to Prof. Alessio Sacco for all the support, availability and help provided whenever I ran into a trouble spot or had a question about this project. This thesis work, accounts for me not only the end of my university career, but also the end of a very long and intense battle, during which I have learned many things and above all, I have changed a lot. For this reason, it is my duty to express my deep gratitude to my family members, family Bonde and Nzeka, and my closest friends for providing me with unfailing support and continuous encouragement in the most difficult moments and through the process of researching and writing this thesis. Without their support, it would not have been possible to complete this accomplishment. Thank you all.

Table of Contents

List of Tables

List of Figures

Acronyms

AI Artificial intelligence

HMM Hidden Markov model

SVM Support vector machines

ARIMA Autoregressive integrated moving average

DT Decision-tree

KNN K-nearest neighbors

NB Naive bayes

BW The Baum-Welch Algorithm

E-step Expectation-step

M-step Maximization-step

Chapter 1

Introduction

Telepathology is the transmission between two distant locations of digital histological images based on a glass slide using telecommunication technologies for research, diagnosis and education purposes. Telecommunications technologies can cause delays which lead to packet loss and difficulties to control the remote device. These difficulties have arisen in a microscope teleoperation system. Teleoperation allows a user to physically control remote operation thanks to commands sent via a telecommunication channel. Nevertheless, telepathology favors an increase in the costs of experts and that IT and telecommunication costs fall, as a consequence, telepathology experts could start to regularly use remote-controlled equipments to the diagnosis. From a location, a pathologist could take control of a physically remote microscope and selects the video images to analyze, then performs the diagnostic of a patient who is in remote rural areas. Therefore, this process can be carried out in one trip to a nearest rural hospital and avoid going to a distant health center, which is often required more than once, for instance, for an initial assessment and another investigation, on the date of the biopsy procedure and later for the results and ancillary procedures before final treatment. However, as its use around the world has become increasingly popular, telepathology is currently dealing with a growing demand to enhance patient safety, quality, reliable consultations, delay and accuracy of diagnostic, etc.

This thesis aims to propose an approach that uses a machine learning (ML) classifierl, Hidden Markov Model (HMM), to predict the future packet content from the network traffic generated during the execution of the LiveMicro system (telepathology session), then, the predicted packet content will be used to remedy packet loss issue and maintain an acceptable delay of data transfers during a live session. LiveMicro is a telepathology system that enables remote computations and consultations. The thesis work is also focused on the comparison between HMM and others ML tecniques such as Support Machine Vectors (SVM), Autoregressive

Integrated Moving Average (ARIMA), Decision-Tree (DT), etc.

The rest of the paper is structured as follows:

- In Section 2: we discuss related telepathology solutions and machine learning models used for the prediction.
- In Section 3: we briefly describe the LiveMicro system and our proposed classifier Hidden Markov Model (HMM).
- In Section 4: we present the structure of our system.
- In Section 5: we descrive the experimental setup and results obtained by measuring performance metrics of HMM and others classifiers.
- In Section 6: we conclude our thesis work.

Chapter 2

Related work

In this section, we will list and describe some systems and applications used in telepathology. Dynamic telepathology involves the transmission and viewing of live images in real time. The most widely used system is a local camera connected to a microscope that provides a live transmission of video-style moving images, viewed over an Internet connection by the remote pathologist on a computer or mobile device. This mode of operation is highly dependent on the operator's ability at the site to locate relevant cells so that the most relevant areas can be viewed by the remote pathologist. An important component of the type of telecopathology is Two-way verbal communication. Ideally, it should be hands free and allow the use of headphones to ensure some privacy when communicating between sites, as shown in [2]. A system that facilitates a faster information exchange, particularly under critical transmission, is shown in [3]. The system proposes a model of Specialized Telepathology Electronic Patient Record (STEPR), exploiting the features (compression efficiency, multiple decompression spatial resolution, region of interest and signal to noise ratio scalability) of emerging image compression standard JPEG 2000. However, the main focus of the model (STEPR based on JPEG 2000) is to offer the flexibility for pathological information manipulation, transmission and processing without compromissing its standardization. In [4] the digital images are uncompressed and shared in high-quality full-resolution. The authors use the new technology of the microscope-integrated telepathology systems. The system exploits an optical module installed on the microscope in order to capture a live feed of the tissue passed on the eyepiece of the microscope. Then, the optical module enables remote participants to view and discuss in real-time a live image that is of high digital pathology grade. During the telepathology session, the remote viewers could perform annotations from their computer screens and, the main microscope user could see annotations on top of the tissue within the microscope eyepiece.

The last section examines different Telepathology solutions and the way in

which data is shared among users over networks. However, this section explores different methods used for the network data prediction. For example in paper [5], the authors presented a long short-term memory (LSTM) neural network model combined with deep neural networks (DNN) to predict network traffic that behaves as a nonlinear system. According to characteristics of autocorrelation, an autocorrelation coefficient is added to the model to improve the accuracy of predictions. The main novelty of the method is to include autocorrelation of the time series in the input of the machine learning algorithm, which leads to high performance with respect to existing methods. Similar approach can be found in [6]. In fact, the authors of [6] introduced NeuTM, a framework for network Traffic Matrix (TM) prediction based on Long Short-Term Memory Recurrent Neural Networks (LSTM RNNs). TM prediction is defined as the problem of estimating future network traffic matrix from the previous ones. In [7] the authors proposed Autoregressive Moving Average (ARMA) time series approach to predict the number of network traffic packets during the next five steps. File Transfer Protocol (FTP) was used to send and receive large files over the network. They capture and collect three hour of network packets at different time intervals, exploiting the network protocol analyzer (Wireshark). The number of packets at each interval is normalized into a log return value, that reflects the relationship between two consecutive network data. During their experiment, they transmit two files of 1GB and 10MB over the network, then used ARMA model which will create the appropriate network traffic pattern. This pattern, together with the log return values, is used to predict the number of network traffic packets during the next five steps. Their result showed that breaking a large file into small size is more efficient and effective than large file for network traffic prediction. Zhou et al. [8], proposed a network traffic prediction based on time series forecasting model ARFIMA, which is an improved ARMA model, to predict real trace records and netflow sampling flow records. In experiment, they combined the data collected from network traffic of CERNET backbone and the ARFIMA model. The result showed that, compare to the ARMA model, the ARFIMA can get more effective network traffic prediction. To overcome the problem of non-liner classification and regression, Weidong et al. [9], introduced a new machine learning method LS-SVM (Least Squares SVM), which is an improved model of SVM (Support Vector Machines). Exploiting NS2 simulator, they simulated the process of the network working with RED and Drop-tail controller, then collected on the bottleneck router the traffic data predicted. The experiment results on the accuracy of prediction was good and feasible.

Due to its good performance in statistics, HMM (Hidden Markov Model) technique is recentlly developed and applied in fields as voice recognition, security situation prediction, network traffic prediction, classification, telemedicine, intrusion detection, etc. In the field of security situation prediction, Wei Liang et al.

[10] proposed the application of the weighted HMM based algorithm to predict the security situation of mobile networks, where the multiscale entropy is used to address the low speed of data training in mobile network, whereas the parameters of HMM situation transition matrix are also optimized. Furthermore, the correlation coefficients are considered as weights, so it can reasonably use the association between the characteristics of the historical data to predict future security situation. In the experiment, DARPA2000 (an offline intrusion detection datase) was used, and with the multiscale entropy method, an appropriate scale factor is selected as training data to train the parameters in HMM model. Finally, the experimental results have shown that the proposed algorithm is highly competitive, with good performance in prediction speed and accuracy when compared to existing security situation prediction techniques. In the field of network traffic prediction [11], the authors provided a study based on Hidden Markov Model (HMM) for the Internet traffic in the Next Generation Mobile Networks (NGMN) such as transitioning 5G and LTE derived using realistic traffic traces collected from various datasets. The model is used to forecast the Quality of Service (QoS) parameters for the given wireless networks. Firstly, they used the model to predict the joint distribution of Inter-Packet Delay Variation (IPDV) and End-to-End Delay (d), then forecast the QoS parameters for the given wireless networks. The simulation is set-up for the LTE and 5G datasets with different sample size and tolerance limit. The results showed that the end-to-end delay and the Inter-packet Delay Variation (IPDV) can be modeled using an HMM with accuracy up to 80% for a percentage tolerance of 10%. In the field of telemedicine [12], the authors used multiple sensors for wearable devices and, integrated the data from different sensors. They introduced an approach based on Hidden Markov Models (HMMs), that analyses the data from each sensor to predict the risk for different diseases that a patient might have. Then, they exploited a rule-based engine that uses the predictions from the HMMS in order to diagnose the health state of a patient for wearable device. In their experiment, they found that a sensor malfunctioning can lead to conflicting results. However, they propose a model of an autonomic system to try to overcome that difficulty. The goal of their research was to detect a specific disease (using HMMs) that differs to the goal of this thesis, which is the prediction of the network packet using HMM.

In this paper, a telemedicine application is used to send and receive data over the network. We exploit the network protocol analyzer (Wireshark) to capture the network packets, then apply HMM to predict the next network packet from the previous one.

Chapter 3

Proposed methods

To achive our goals, we apply the machine learning model for prediction during the execution of LiveMicro application. This chapter deals with the theory behind hidden Markov models and their main problems, along with a brief description of the LiveMicro.

3.1 LiveMicro

LiveMicro is a telepathology system that enables remote computations and consultations. Remote consultation allows a telepathologist to access a live telepathology session and remotely manipulate the virtual instance of the microscope. Pathologists usually carry glass slides in order to analyze histological samples. This latter allows to minimize the diagnosis time.However, LiveMicro eliminates the necessity to transfer physical slides and allows a remote specialists to consult at distance. This preserves time and unnecessary expense, resulting in better patient care and faster diagnoses. It allows services such as focus of a microscope, zooming and remote control of panning on *software-defined glass slides*. In our thesis work, LiveMicro is used in order to perform a live session and generate datasets.

3.2 Markov Model

The Markov model is a stochastic model used to model temporal or sequential data. It is assumed that future states depend only on the current state, not on the events that occurred before it (Markov property). Formally, a Markov model is specified by the set of states (Q), a transition probability matrix (A) and an initial probability distribution over states (π). In the Markov model, each state corresponds to an observable event.

3.3 Hidden Markov Model

A hidden Markov model (HMM) is a Markov model that exploits both observed events and hidden events. In hidden Markov model defined above and illustrated in Figure 3.1, the states are considered as hidden events. Thus, they are not directly visible to the observer. However, only the output symbols (also called observations) can be observed.

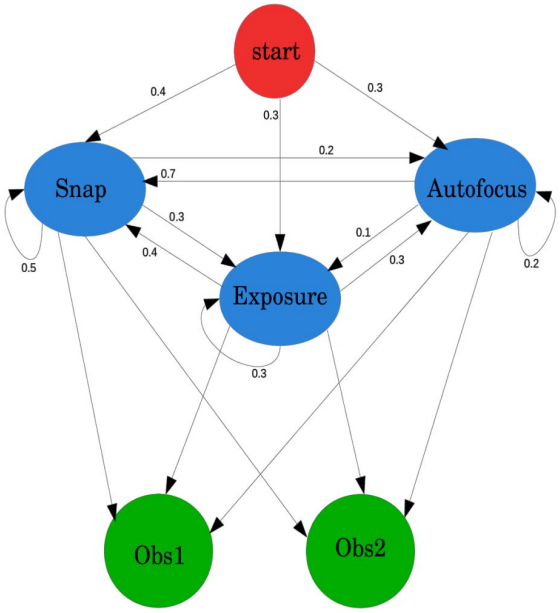

Figure 3.1: A Hidden Markov Model showing the initial states probabilities, transition probabilities and emission probabilities.

The hidden Markov model is composed of the following five elements:

1. **A set of states** denoted as

$$Q = \{q_1, q_2, \ldots, q_N\}, \tag{3.1}$$

where N representing the number of states in the model. In our sytem, we consider the field "ActionType" (action that a pathologist performs on the screen in a telepathology session) of the dataset as state, i.e, Snap, Exposure, Autofocus, etc.

2. **A transition probability matrix** over states,

$$A = \{a_{ij}\}_{NxN}, \quad a_{ij} = P(q_j^{t+1} \mid q_i^t), \quad \text{for} \quad t = 1,2,...,T-1 \tag{3.2}$$

where a_{ij} expressing the probability of moving from state i to state j.

3. **A sequence of observations**,

$$V = \{v_1, v_2, \ldots, v_M\}, \tag{3.3}$$

where M is the number of observations values in the model. For each entry in the dataset, we use the following set of fields as an observation: Source IP, Destination IP, Length and Ratio (request/response) size.

4. **A sequence of observation likelihoods**, also called emission probability matrix,

$$B = \{b_{jk}\}_{NxM}, \quad b_{jk} = P(v_k^t \mid q_j^t), \quad \text{for} \quad t = 1,2,...,T \tag{3.4}$$

where b_{jk} representing the probability of an observation v_k being generated from a state j at time t.

5. **The initial states probabilities** over states,

$$\pi = \{\pi_1, \pi_2, \ldots, \pi_N\}, \quad \pi_i = P(q_i^t), \quad \text{for} \quad t = 1 \tag{3.5}$$

where π_i defines the probability to start in state q_i.

Finally, hidden Markov model can be represented as λ = (A, B, π). There are three basic problems for HMMs:

- **First problem (Likelihood):** Given the observation sequence in time O = (o_1,o_2,...,o_T) and a HMM model λ = (A, B, π), determine the likelihood P(O|λ);
- **Second problem (Decoding):** Given the observation sequence in time O = (o_1,o_2,...,o_T) and a HMM model λ = (A, B, π), find the most likely state sequence Q = (q_1,q_2,...,q_T);
- **Third problem (Training):** Given the observation sequence in time O = (o_1,o_2,...,o_T) and the set of states in the HMM, find the model λ = (A, B, π) that maximizes the probability of O.

3.3.1 Likelihood problem: The Forward Algorithm

Given an HMM with an observation sequence of T observations and N hidden states, it requires N_T possible hidden sequences. In real-world applications, the direct computation of P(O|λ) is hard and sometimes impossible, because N and T both are wide numbers, so N_T will be very wide. The more efficient way to solve P(O|λ) is the recursive forward algorithm. The parameters α_t(i) are keys elements

of the algorithm, and defined as the joint probability of observing the sequence up to time t and the Markov process being in state s_t at time t:

$$\alpha_t(i) = P(o_1, o_2, \ldots, o_t, s_t = q_i|\lambda) \tag{3.6}$$

The computation process:

1. **Initialization:**
$$\alpha_1(i) = \pi_i b_i(o_1); \quad 1 \leq i \leq N$$

2. **Recursion:**
$$\alpha_t(i) = \sum_{i=1}^{N} \alpha_{t-1}(i) a_{ji} b_i(o_t); \quad 1 \leq i \leq N, \quad 1 < t \leq T$$

3. **Termination:**
$$P(O \mid \lambda) = \sum_{i=1}^{N} \alpha_T(i)$$

3.3.2 Backward algorithm

Given the model λ and we are in state s_i at time t, the backward variable is the probability of the partial observation sequence from *t + 1* to the end:

$$\beta_t(i) = P(o_{t+1}, o_{t+2}, \ldots, o_T \mid q_t = s_i, \lambda) \tag{3.7}$$

The computation process is approximately similar to the Forward algorithm:

1. **Initialization:**
$$\beta_T(i) = 1; \quad 1 \leq i \leq N$$

2. **Recursion:**
$$\beta_t(i) = \sum_{j=1}^{N} \beta_{t+1}(j) a_{ij} b_j(o_{t+1}); \quad 1 \leq i \leq N, \quad 1 \leq t < T$$

3. **Termination:**
$$P(O \mid \lambda) = \sum_{j=1}^{N} \pi_j b_j(o_1) \alpha_1(j)$$

3.3.3 Decoding problem: The Viterbi Algorithm

Decoding problem is a very common situation regarding the hidden Markov models, the idea is to predict the generating state path of a particular observed sequence. This problem could be solved using both Forward and Backward algorithms. Given both the model λ and the observation sequence, the most likely state s_i at time t can be obtained through the γ_ti parameter, defined as:

$$\gamma_t(i) = P(q_t = s_i \mid O, \lambda) \tag{3.8}$$

As the Forward parameter (α_t(i)) evaluates the state probability before time t and, the backward parameter (β_t(i)) takes care of the state probability after time t, so, the γ parameter can be calculated using Froward and Backward parameters as:

$$\gamma_t(i) = \frac{\alpha_t(i)\beta_t(i)}{P(O \mid \lambda)} \tag{3.9}$$

Thus, the most likely state at time t can be considered as the state q_t for which γ_t is higher:

$$q_t = \underset{1 \leq i \leq N}{\operatorname{argmax}} \left[\gamma_t(i)\right], \quad 1 \leq t \leq T \tag{3.10}$$

However, this approach of individually most likely state presents a huge inconvenience. For instance, given a model that holds unfeasible transitions, i.e., the transition from state i to j is zero ($a_{ij} = 0$); the resulting sequence of states may not be generated by the model whether the sequence contains those transactions. This problem can be solved by using a standard algorithm, called *Viterbi algorithm*, that uses a dynamic programming approach in order to maximize P[S | O, λ] (the likelihood of the whole generating state sequence). Finally, we can compute the Viterbi recursion as follows:

1. **Initialization:**

$$\delta_1(i) = \pi_i b_i(O_1); \quad 1 \leq i \leq N$$
$$\psi_1(i) = 0; \quad 1 \leq i \leq N$$

2. **Recursion:**

$$\delta_t(i) = \max_{1 \leq i \leq N} \left[\delta_{t-1}(j) a_{ji}\right] b_i(O_t); \quad 1 \leq i \leq N, \quad 1 < t \leq T$$

$$\psi_t(i) = \underset{1 \leq i \leq N}{\operatorname{argmax}} \left[\delta_{t-1}(j) a_{ji}\right]; \quad 1 \leq i \leq N, \quad 1 < t \leq T$$

3. **Termination:**

$$\textbf{The best score: } P^* = \max_{i=1}^{N} \delta_T(i)$$

$$\textbf{The start of backtrace: } q_T^* = \underset{i=1}{\overset{N}{\operatorname{argmax}}} \delta_T(i)$$

3.3.4 Training probelm: The Baum-Welch Algorithm

Now, let us look at the HMM training problem (also called learning problem): Given an observation sequence O and knwon number of states, the training process aims to find the optimal model parameters, $\lambda = (\mathrm{A}, \mathrm{B}, \pi)$, that maximizes the probability of the observation sequence, $\mathrm{P} = (O \mid \lambda)$. The standard algorithm to solve this problem is the *Baum-Welch Algorithm* (forward-backward algorithm), that is an instance of a general family of *EM* (expectation-maximization) algorithms. Thus, the Baum-Welch Algorithm starts by iterativelly estimating the initial transition, emission and state start probabilities, then using those estimates to computes better and better estimates. In practice, the Baum-Welch process requires initial values for the model parameters $\lambda = (\mathrm{A}, \mathrm{B}, \pi)$, which will have the influence on the performace. In the E-step, the algorithm computes $\xi_t(\mathrm{i,j})$ (joint probabilities) and $\gamma_t(\mathrm{i})$ (di-gamma) variables:

$$\xi_t(i) = P(q_t = s_i, q_{t+1} = s_j \mid O, \lambda) = \frac{\alpha_t(i) a_{ij} b_j(o_{t+1}) \beta_{t+1}(j)}{P(O \mid \lambda)}, \tag{3.11}$$

$$\gamma_t(i) = P(q_t = s_i \mid O, \lambda) = \sum_{j=1}^{N} \xi_t(i, j), \tag{3.12}$$

where $\xi_t(\mathrm{i})$, as shown in Figure 3.2, is the probability of being in state s_i at time t and state s_j at time $t + 1$, given the model and the observation sequence, while $\gamma_t(\mathrm{i})$ is defined as the probability of being in state s_i at time t.

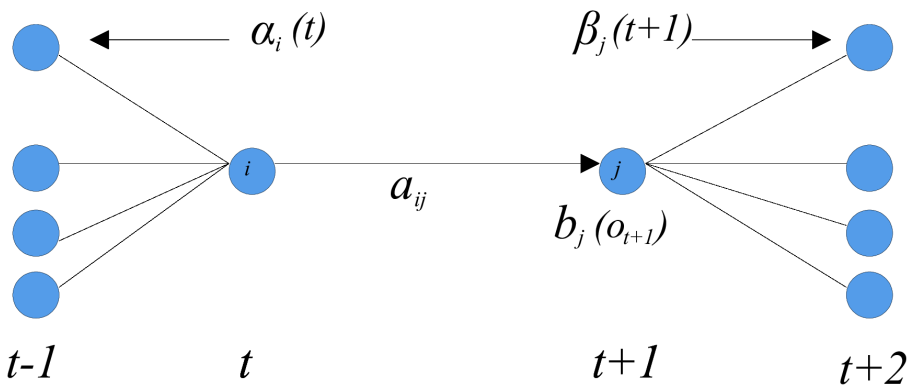

Figure 3.2: Calculation of the joint probability of being in state i at time t and state j at time $t + 1$.

Finally, the M-step performs reestimation of the model parameters in order to improve them:

1. **State transition propability:**

$$\bar{a}_{ij} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)}; \quad 1 \leq i \leq N, \quad 1 \leq j \leq N, \tag{3.13}$$

   where $\bar{a}_{ij}$ is expressed as the expected number of transitions from state i to state j divided by the expected number of transitions from state i.

2. **Emission probability:**

$$\bar{b}_j(k) = \frac{\sum_{t=1, o_t=v_k}^{T} \gamma_t(j)}{\sum_{t=1}^{T} \gamma_t(j)}; \quad 1 \leq j \leq N, \quad 1 \leq k \leq M,$$

   where $\bar{b}_j(k)$ is ythe expected number of times in state j and observing symbol v_k divided by the expected number of times in state j.

3. **Initial state probability:**

$$\bar{\pi}_i = \gamma_1(i); \quad 1 \leq i \leq N,$$

   where $\bar{\pi}_i$ is equal to the probability of being in state i at time *1*.

Chapter 4

Architecture

As practicing pathology at a distance requires the use of microscope, we consider the LiveMicro application together with the Hidden Markov Model as one system. In Figure 4.1, we see a general overview of the system. LiveMicro represents a virtual microscope that allows to perform a live telepathology session. Defined as *Data collector*, it collects network data during a live session and prepares them for the preprocessing part. The preprocessing step, analyzes the dataset selecting the information to be considered as observations and hidden states. This part also splits the data into training and test sets. Thus, the training set is used in order to train our model using the *Baum–Welch algorithm*, then, create a new HMM model for the test part. However, the test part is done on the test set, and it is divided into two parts: first, we predict the sequence of hidden states using the *Viterbi algorithm*, given the test sequence of observations. Secondly, for each state (at time t) in the predicted sequence, we exploit the transition probability matrix in order to find the next state at time $t+1$.

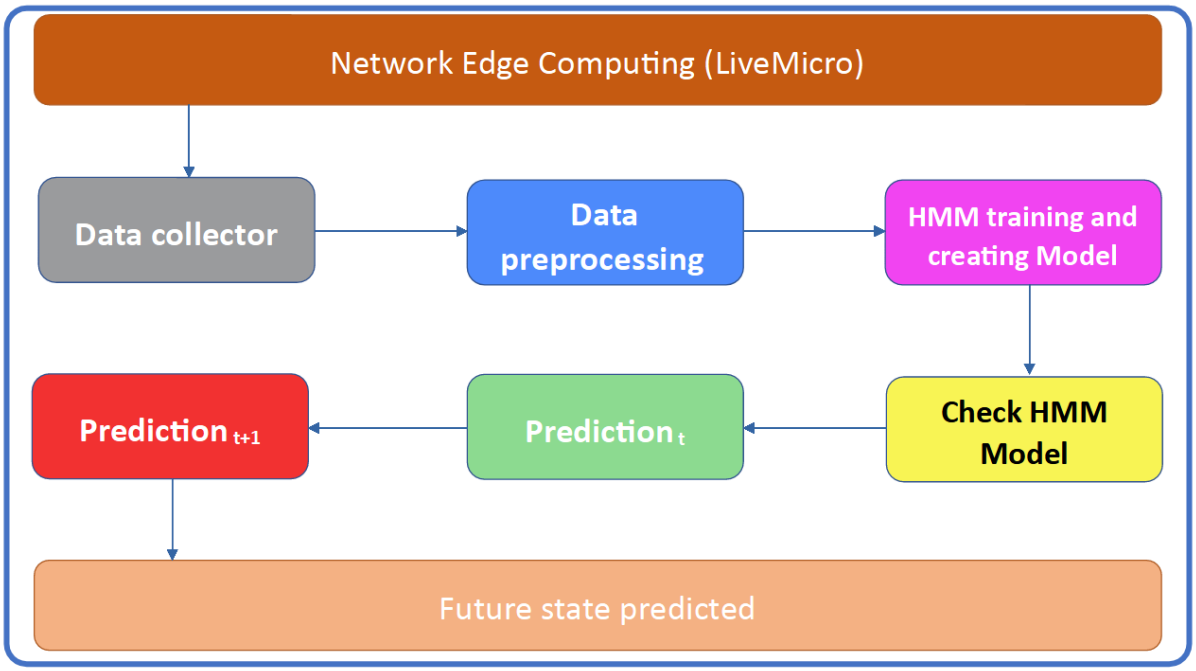

Figure 4.1: Detailed view of the system

The process of data recovery can be adopted when observing the phenomenon of packet loss. This process involves packet retransmissions technique in order to recover packet losses. Thus, in this work, the next state predicted is used to reconstruct the original data when occur packet losses, and retransmit them during the execution of the telepathology session.

Chapter 5

Experimental setup and results

This chapter presents the way our model was trained and tested, along with the measurements carried out in order to measure the performance of the model in terms of accuracy, precision, recall, F1-Score and training time. First, we introduce the training and testing data used in our system. Then, we present the evaluation of our model and the comparison with others ML models.

5.1 DataSet

In this project, we used different datasets to test our model. The datasets, considered as *scenario1* and *scenario2*, are generated during the execution of LiveMicro application. Both scenario1 and scenario2 represent the set of packets exchanged during a telepathology session. Scenario1 contains three types of states (classes), while scenario2 has two, as shown in Figure 5.1. Each class corresponds to the packet content, which is the action types executed by the pathologist.

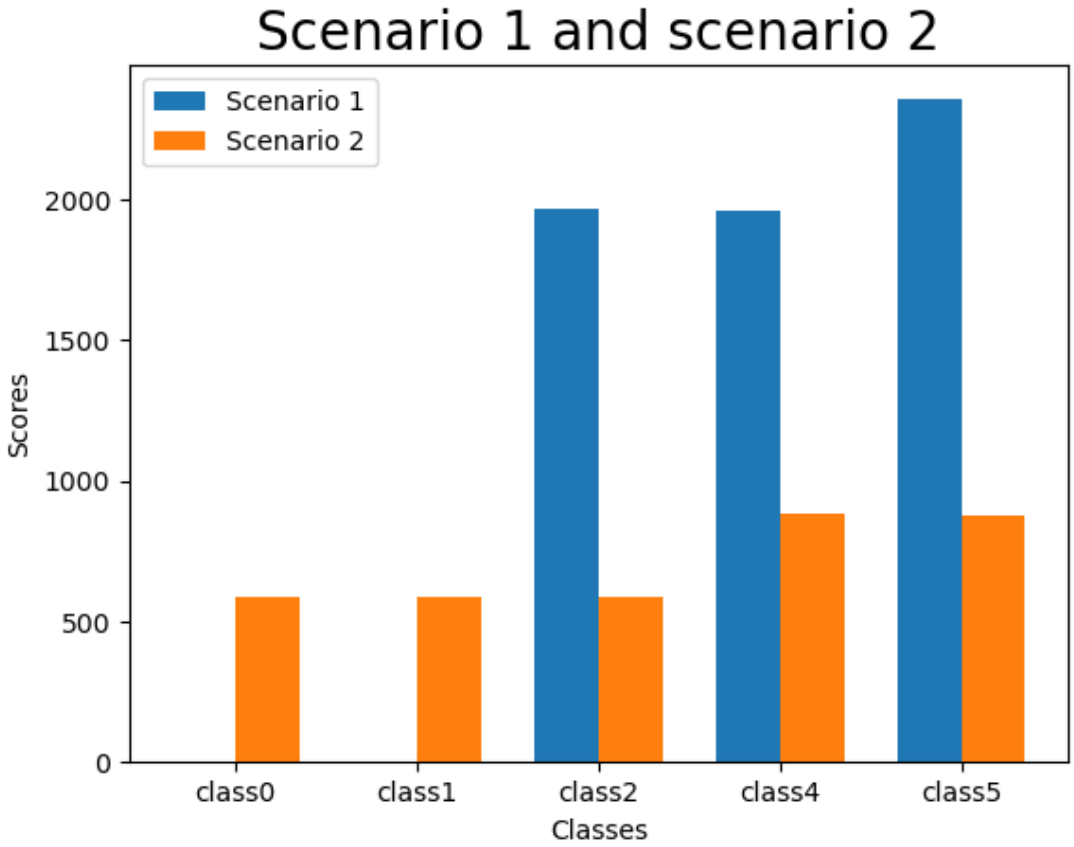

Figure 5.1: State distribution histograms in scenario1 and scenario2.

5.2 Experimental setup

Evaluation of the Hidden Markov Model requires initial values for the model parameters, i.e., state initial probabilities, transition matrix over states and emission probabilities. In practice, it is important to have good initial parameters for HMM to achieve accurate classification results in a reasonable time. In this work, we calculated the initial parameters by randomly generating the values until we found those that yield an acceptable result.

For the training and testing phase, we splitted the dataset into train (80%) and test (20%) sets. This is because the model will use the data in the training set to learn HMM parameters, then use the learned parameters to make prediction about the data in the test set.

5.3 Performance Metrics

The performance evaluation of the ML model is an essential part of any project. The choice of metrics affects how the performance of ML model is evaluated and compared. In most cases, the classification accuracy is used to measure the model performance, but it is not sufficient to truly judge the model. In this work, we compare the performance of the classification models according to certains metrics like training time, precision, accuracy, recall and f1-score. However, the metrics can be calculated using a confusion matrix:

1. **Confusion Matrix:** It is used to measure the performance of a classifier on a set of test data whose true values are known. A confusion matrix is a table of two dimensions, *Predicted* and *Actual*, in which each dimension contains the following elements: True Positives (TP), True Negatives (TN), False Positives (FP), False Negatives (FN), as shown in Figure 5.2.

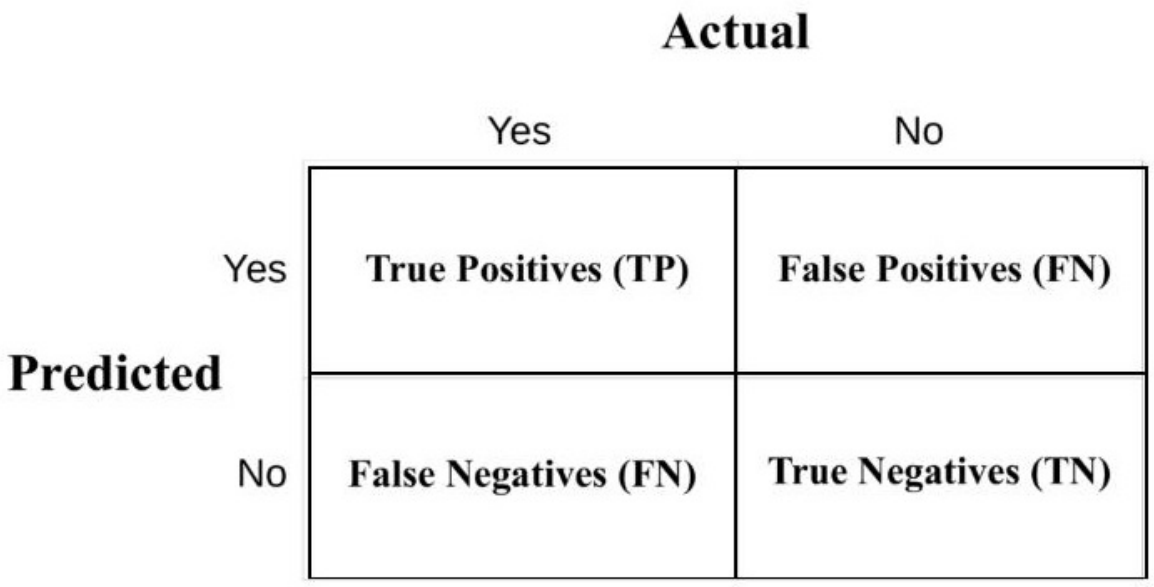

Figure 5.2: A Confusion Matrix.

Where TP means both predicted class and actual class of data is *Yes*, while TN occurs when both predicted class and actual class of data is *No*. FP means predicted class of data is *Yes* and actual class of data is *No*, and FN when predicted class of data is *No* and actual class of data is *Yes*.

2. **Accuracy:** It expresses the percentage (0% - 100%) of the exact predicted class in total number of instances. The Accuracy can be calculated as:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{5.1}$$

3. **Precision:** It quantifies the number of positive class predictions that actually belong to the positive class. The Precision can be calculated as:

$$Precision = \frac{TP}{TP + FP} \tag{5.2}$$

4. **Recall:** It quantifies the number of positive class predictions made out of all positive examples in the dataset. The Recall can be calculated as:

$$Recall = \frac{TP}{TP + FN} \tag{5.3}$$

5. **F1-score:** This score refers to the harmonic average of the Recall and Precision metrics, i.e., it is the weighted average of the Recall and Precison metrics. It varies from 0 (worst) to 1 (best). The F1-score can be calculated as:

$$F1 - score = \frac{2 * Precision * Recall}{Precision + Recall} \tag{5.4}$$

6. **Training time:** The time (usually expressed in seconds) that a classifier takes for building a model on the set of training data. The lower the value of this time, the fitter will be the model.

We exploited the function *Classification report* of the library *sklearn.metrics* to obtain the classification report of the model.

5.4 Results

In this section we present some experimental results achieved with the our proposed classification model shown in Section 3.3 and the comparison with others classifiers, i.e., Generative adversarial networks (GAN), Support Machine Vectors (SVM), Autoregressive Integrated Moving Average (ARIMA), Decision-Tree (DT), K-Nearest Neighbors (KNN), Naives-Bayes (NB). Table 5.1 and 5.2 summarize the performance metrics of all the classification models, for the scenario1 and scenario2 respectively. Figure 5.3 illustrates the comparative analysis of different models with regard to their accuracy, precision, recall and F1-score for the scenario1. From the chart, we can notice that the accuracy is highest for HMM (87.28%), and lowest for

ARIMA (43.72%). SVM achieves the accuracy of 62.48%, while GAN gets 69.57%. The remaining models have the same accuracy of 68.76%. HMM obtains the same and maximum value (0.87) of precission and recall, while ARIMA achieves the same and maximum value (0.44) of precission and recall. The F1-scorel is maximum for HMM (0.87) and minimum for ARIMA (0.41). Figure 5.4 shows the comparative analysis of different models with regard to their accuracy, precision, recall and F1-score for the scenario2. The accuracy is highest for HMM (82.84%), and lowest for ARIMA (8.37%). ARIMA presents the same and lowest value (0.08) of precision, recall and F1-score, while HMM obtains 0.72, 0.83 and 0.76, for precision, recall and F1-score respectively. Figure 5.5 shows the training time chart of different machine learning models for the scenario1. GAN consumes the highest time of 525.73 seconds, however DT and NB consume the lowest time of 0.01 seconds. Figure 5.6 shows the training time chart for the scenario2. GAN consumes the highest time of 304.11 seconds, while DT and NB consume only 0.01 seconds.

Table 5.1: The performance metrics of all the classifiers (scenario1).

	Accuracy (%)	**Precison**	**Recall**	**F1-score**	**Training time (s)**
HMM	87.28	0.87	0.87	0.87	0.45
ARIMA	43.72	0.44	0.44	0.41	172.71
SVM	62.48	0.53	0.69	0.58	0.23
DT	68.76	0.53	0.69	0.58	0.01
NB	68.76	0.53	0.69	0.58	0.01
KNN	68.76	0.53	0.69	0.58	0.06
GAN	69.57	0.61	0.73	0.67	525.73

Table 5.2: The performance metrics of all the classifiers (scenario2).

	Accuracy (%)	**Precison**	**Recall**	**F1-score**	**Training time (s)**
HMM	82.84	0.72	0.83	0.76	0.51
ARIMA	8.37	0.08	0.08	0.08	74.33
SVM	58.44	0.52	0.67	0.57	0.07
DT	66.52	0.52	0.67	0.57	0.01
NB	66.52	0.52	0.67	0.57	0.01
KNN	66.52	0.52	0.67	0.57	0.02
GAN	58.68	0.51	0.59	0.54	304.11

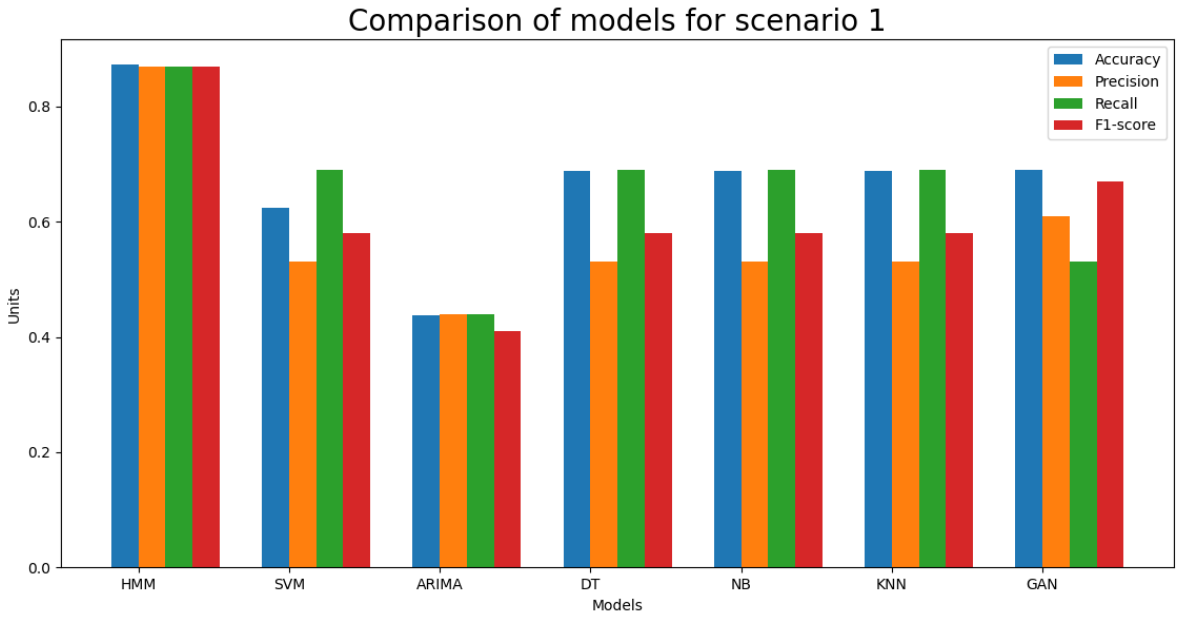

Figure 5.3: The comparison of models in terms of accuarcy, precision, recall and F1-score (scenario1).

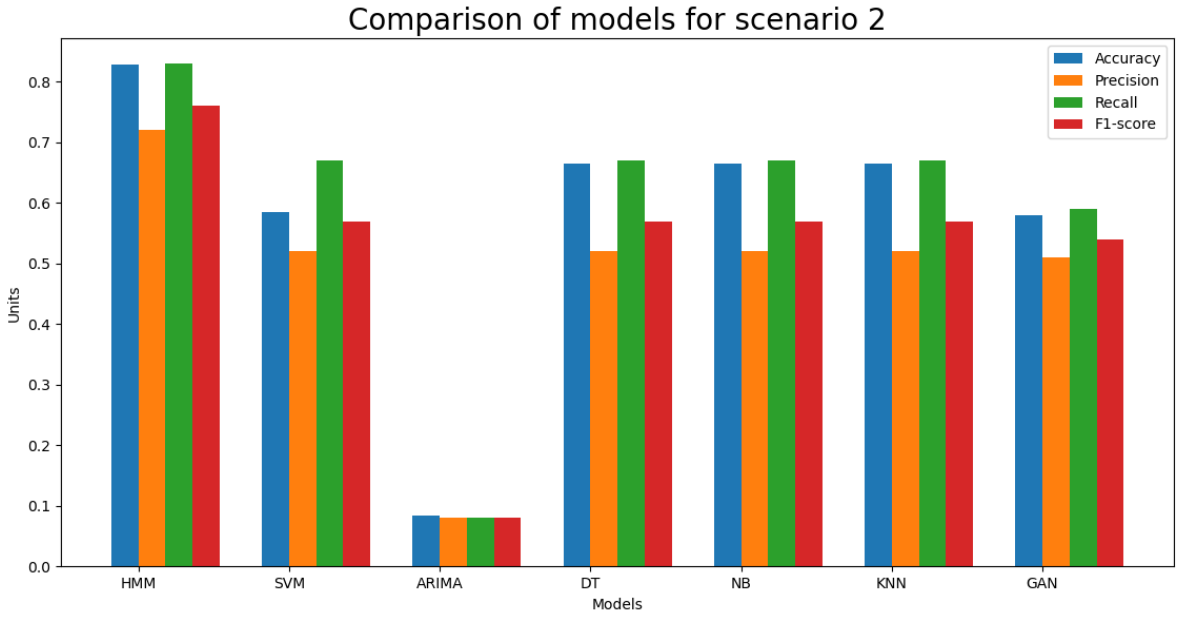

Figure 5.4: The comparison of models in terms of accuarcy, precision, recall and F1-score (scenario2).

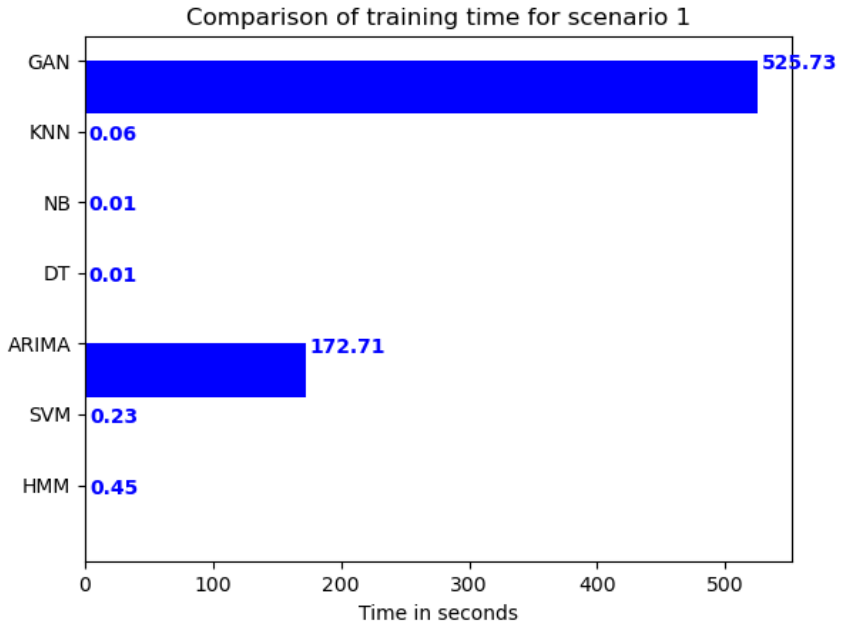

Figure 5.5: The comparison of models in term of training time (scenario1).

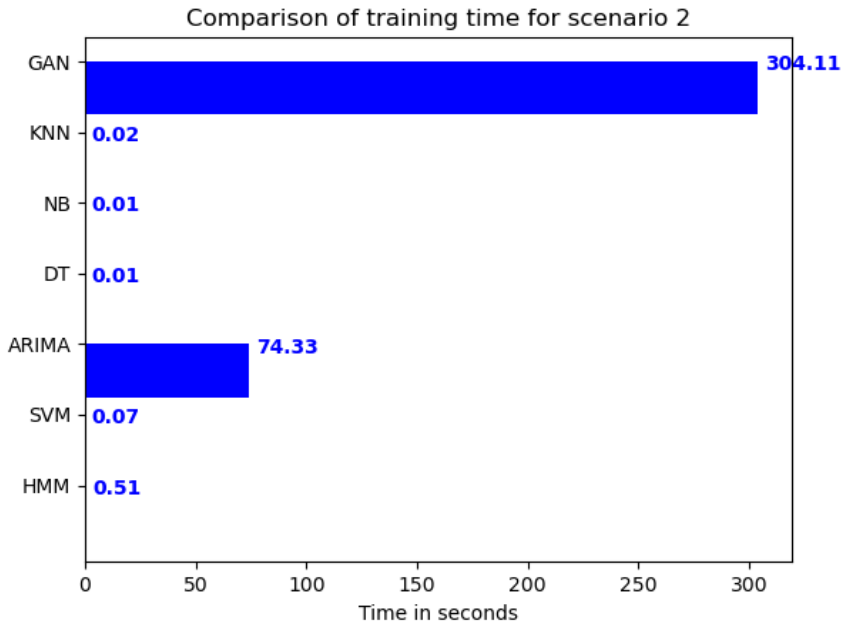

Figure 5.6: The comparison of models in term of training time (scenario2).

In summary, these findings indicate that the proposed method, HMM, is the best machine learning algorithm analyzed for this study. This model has been shown to be effective in accurately predicting the content of network packets in order to overcome both packet loss and time delay issues during the telepathology session.

Bibliography

[1] Sacco Alessio et al. «*On Edge Computing for Remote Pathology Consultations and Computations*». In: *IEEE Journal of Biomedical and Health Informatics* 24.9 (2020), pp. 2523–2534 (cit. on p. i).

[2] Collins B.T. «*Telepathology in Cytopathology: Challenges and Opportunities*». In: *Acta Cytologica 2013* (2013), pp. 221–232 (cit. on p. 3).

[3] L. Montesinos and J. Puentes. «*Specialized telepathology electronic patient record based on JPEG 2000*». In: *4th International IEEE EMBS Special Topic Conference on Information Technology Applications in Biomedicine* (2003), pp. 110–113 (cit. on p. 3).

[4] Siegel et al. «*New Technologies: Real-time Telepathology Systems—Novel Cost-effective Tools for Real-time Consultation and Data Sharing*». In: *Toxicologic Pathology* 45 (2017), pp. 1039–1042 (cit. on p. 3).

[5] Wang Shihao et al. «*Long short-term memory neural network for network traffic prediction*». In: *ISKE* (2017), pp. 1–6 (cit. on p. 4).

[6] A. Azzouni and et al. «*Neutm: A neural network-based framework for traffic matrix prediction in sdn*». In: *CoRR* (2017) (cit. on p. 4).

[7] PN. K. Hoong et al. «*Impact of Utilizing Forecasted Network Traffic for Data Transfers*». In: (), p. 211 (cit. on p. 4).

[8] Zhou et al. «*Network traffic prediction based on ARFIMA model*». In: *International Journal of Computer Science Issues* 9 (2013) (cit. on p. 4).

[9] Weidong Luo, Xingwei Liu, and Jian Zhang. «*SVM-based analysis and prediction on network traffic*». In: *International Conference on Intelligent Systems and Knowledge Engineering 2007* (2007), pp. 1951–6851 (cit. on p. 4).

[10] Wei Liang et al. «*A Security Situation Prediction Algorithm Based on HMM in Mobile Network*». In: *Wireless Communications and Mobile Computing* 2018 (2018), p. 11 (cit. on p. 5).

[11] Dash et al. «*Traffic Prediction in Future Mobile Networks using Hidden Markov Model*». In: (2019) (cit. on p. 5).

[12] Neyens, Gilles, and Zampunieris. «*Using Hidden Markov Models and Rule-based Sensor Mediation on Wearable eHealth Devices*». In: *11th International Conference on Mobile Ubiquitous Computing, Systems, Services and Technologies* (2017), p. 11 (cit. on p. 5).