Fast Detection of Specific Information in Voice Signal over Internet Protocol

HAZEM M. EL-BAKRY

Faculty of Computer Science & Information Systems,
Mansoura University, EGYPT
E-mail: helbakry20@yahoo.com

NIKOS MASTORAKIS

Department of Computer Science,
Military Institutions of University Education
(MIUE) -Hellenic Naval Academy, Greece

Abstract: The world Telecommunications industry caters to both Wireline and Wireless business. In the present Fourth Generation (4G) wireless world rapid change is the only constant factor. The biggest paradigm change being the move from Circuit Switched to Packet Switched technologies. Voice over Packet (VoP) is transforming telephony by rapidly replacing the legacy circuit-switched technology. The VoP technology unites the telephony and data worlds allowing voice, facsimile, and video traffics to be relayed over managed IP networks or corporate intranets around the world. In this paper, various aspects of Voice over Internet Protocol (VoIP) are discussed, including the architecture for the NGMN, challenges, standards and the application areas. In addition a new approach for fast detecting certain information in VOIP is presented. The entire data are collected together in a long vector and then tested as a one input pattern. Proposed fast time delay neural networks (FTDNNs) use cross correlation in the frequency domain between the tested data and the input weights of neural networks. It is proved mathematically and practically that the number of computation steps required for the presented time delay neural networks is less than that needed by conventional time delay neural networks (CTDNNs). Simulation results using MATLAB confirm the theoretical computations.

Keywords: Voice over IP, Next Generation Mobile Networks, Fast Information Detection, Cross Correlation, Frequency Domain

1. Introduction

Voice over IP (VoIP) is a set of technologies that enable voice calls to be carried over the Internet (or other networks designed for data), rather than the traditional telephone landline system—the Public Switched Telephone Network, or PSTN. The potential for very low-cost or free voice calls is driving the use of the technology but in the long-term, VoIP is more significant than just free phone calls - it represents a major change in telecommunications. The fact that VoIP transmits voice as digitized packets over the Internet means that it has the potential to converge with other digital technologies, which in turn will result in new services and applications becoming available. However, the adoption of VoIP is not without complications. The traditional PSTN telephone infrastructure has been built up over the last one hundred years or so and has developed into a robust voice communications system that provides reliability figures of nearly 100%. In contrast, VoIP is a relatively new technology with a fledgling architecture that is built on inherently less reliable data networks. This means that there are therefore justifiable concerns around the extent to which it is deployed.

VoIP has developed considerably in recent years and is gaining widespread public recognition and adoption through consumer solutions such as Skype and BT's strategy of moving to an IP-based network. This adoption is spreading into the F&HE domains, with a number of institutions implementing VoIP, and through work being undertaken by UKERNA. The technology offers opportunities for the development of new applications and educational services, particularly through the potential for converging voice with other media and data. In the long-term, VoIP is likely to impact on some of the bigger picture developments within further and higher education such as virtual universities, identity management, and integration with enterprise-level services and applications.

The term VoIP was coined by the VoIP Forum which was set up in May 1996 as an industry group concerned with promoting and developing product interoperability and a high quality of service for Internet telephony products. Initially, one of the main drivers in developing VoIP was the potential to cut the cost of telephone calls. Traditional voice calls, running over the PSTN, are made using circuit switching, where a dedicated circuit or channel is set up

between two points before the users talk to one another—just like old-fashioned operators, plugging in

the wires to connect two callers. The advantage of this is that once the circuit is set up, the call quality is very good, because it is running over a dedicated line. But this type of switching is expensive because the network needs a great deal of (mostly under-used) capacity. The development of VoIP represents a major change in telecommunications. VoIP uses IP protocols, originally designed for the Internet, to break voice calls up into digital 'packets'. In order for a call to take place the separate packets travel over an IP network and are reassembled at the far end. The breakthrough was in being able to transmit voice calls, which are much more sensitive to any time delays or problems on the network, in the same way as data.

Whereas calls over the PSTN are metered, so the user pays for the amount of time taken by their call, Internet usage is not metered. The user pays a set fee for their Internet service and their VoIP service and can then use the Internet to get free phone calls to other users on the same VoIP service, or pay a small fee to call users on other VoIP services or on the PSTN.

Packetised voice also enables much more efficient use of the network because bandwidth is only used when something is actually being transmitted. Also, the network can handle connections from many applications and many users at the same time, unlike the dedicated circuit-switch approach. This greater efficiency is one of the main reasons that all major carriers, such as BT with its 21CN (21st Century Network) project2, are changing their own networks so that they are IP-enabled.

Packet networks today, are growing at a rate much faster than the voice networks. The main reason behind this tremendous growth of packet networks has been the growth of the Internet. At this growth rate, the amount of data traffic is bound to exceed the amount of voice traffic pretty soon. Of late, there has been a growing interest in transporting voice over packet networks, mainly because of the low-cost offered by these packet networks. VoIP is the transport of voice over the Internet Protocol. In simple terms, it is the ability to make a telephone call over the IP network, at a cost much lower than traditional telephone networks.

Even though VoIP [1,2] presents a tremendous opportunity, the Internet Protocol was not designed to carry voice, and hence providing toll quality voice over IP is a major challenge. Major issues revolve around the quality of the voice calls as well as the ease of use for the end user. Significant progress, however, has been made in this respect to engineer the packet networks to provide toll quality voice. VoIP, hence has a major potential for being a low cost alternative to PSTN. It also has the potential of replacing the telephone network with an integrated network, capable of supporting both voice and data over a common infrastructure.

In this paper, various aspect of VoIP are discussed, including the challenges, application areas, and the standards. Section II describes the advanced architecture for the NGMN. Section III talks about VoIP- related standards and protocols, as defined by ITU-T and IETF. The key standards are H.323, MGCP/MEGACO, and SIP Section IV elaborates upon issues and challenges of VoIP. The key issues are providing Quality of Service (QoS), maintaining network transparency, and providing inter-working between heterogeneous networks. Section V talks about VoIP applications including Internet telephony, VOIP based call-centers, and trunking applications. Section VI concludes the paper.

ARCHTITECTURE

This section describes the advanced architecture of a generic protocol stack for the NGMN. In order to provide an efficient and easily portable stack, we have defined a common design framework for all protocol stacks (wireline and wireless). The salient features of this stack are as follows:

1. *Modularity:* All stacks adopt a layered, modular and scalable architecture.
2. *Dual-interface Approach:* Both Functional as well as Message-based interfaces are supported.
3. *OS independence:* The stacks developed are made independent of target OS. OS dependent code is localized to a particular file ensuring easy portability.
4. *Flexible configuration: S*tacks come along with Compile-time and Run-time support for optional features, including debug-traces, statistics collection and error reporting.
5. *Comprehensive error handling:* To provide comprehensive error handling, the stacks are provided with Multi-level error-reporting and debug-traces facility.
6. *Redundancy support:* All stacks have an in built redundancy support.

2. VOIP Related Standards and Protocols

The standardization activity of VoIP is being governed by two bodies, namely the ITU-T and the IETF. The following sub-sections elaborate on the standardization effort.

ITU-T Standards

The first set of standards related to VoIP was developed by ITU-T through their H.323 series. This standard, along with other standard developed by ITU-T are detailed below.

H.323

H.323 standard provides an infrastructure for audio, video and data communications over packet based networks that may not provide Quality of Service (QoS). The H.323 standard is a part of H.32x protocol family, that includes, besides H.323, standards like H.323 (standard for LANs) and H.320 (standard for ISDNs) among others.

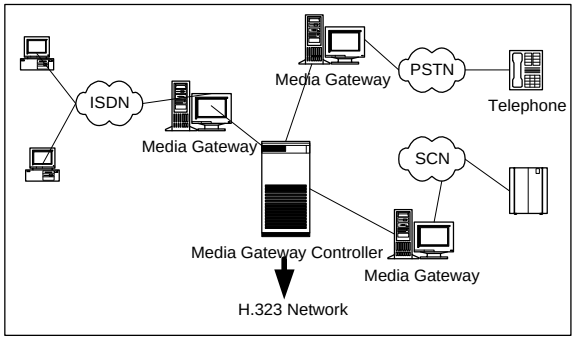

Figure 1. H.323 Network components.

The H.323 standard describes four key components for an H.323 system, namely the terminals, gateways, gatekeepers, and Multipoint Control Units (see Figure 1) components are described in the following sub-sections.

Terminals

A *terminal* is a PC or a standalone device running an H.323 protocol and the multimedia applications. A terminal supports audio communications and can optionally support video or data communications. The primary goal of H.323 is to interwork with other multimedia terminals. Because the basic service provided by an H.323 terminal is audio communications, a H.323 terminal plays a key role in IP-telephony services.

Gateways

A *gateway* connects two dissimilar networks. An H.323 gateway provides connectivity between an H.323 network and a non-H.323 network. This connectivity of dissimilar networks is achieved by translating protocols for call-setup and release, converting media formats between different networks, and transferring information between different networks connected by the gateway. A gateway is not required however for communication between two terminals on an H.323 network. The gateways perform functions like search (conversion of called party phone to IP address), connection, digitization, demodulation, compression/decompression, and demodulation.

Gatekeepers

Gatekeepers can be considered the brain of H.323 network. It is the focal point for all calls within the H.323 network. Although they are not mandatory, they important services like address translation, admission control, bandwidth management, zone-management, and call-routing services.

Multipoint control units (MCU)

These provide support for conferences between three or more H.323 terminals. All terminals participating in the conference establish a connection with the MCU. The MCU manages conference resources, negotiates between terminals for the purpose of determining the audio or video coder/decoder to use, and may handle the media stream.

H.225

H.225 is a standard, which covers narrowband visual telephone services defined in H.200/AV.120 series recommendations. It specifically deals with those situations where the transmission path includes ore or more packet-based network, each of which is configured and managed to provide a non-guaranteed QoS. H.225 describes how audio, video, data and control information on a packet-based network can be managed to provide conversational services in H.323 equipment.

H.248

H.248 is same as MEGACO standard published by IETF, and is discussed in sub-section A2.2.

IETF Standards

Internet Engineering Task Force (or IETF) along with ITUT is playing key role in VoIP-related standardization effort. The following sub-sections elaborate upon key areas and their standards (RFCs).

Media Gateway Control Protocol (MGCP)

The Media Gateway Control Protocol or MGCP [3] implements the interface between a Media Gateway (MG) and a Media Gateway Controller. This interface is implemented as a set of transactions. The transactions are composed of a command and a mandatory response. The MGCP protocol is detailed in RFC2705.

Megaco

Megaco/H.248 is the media gateway control protocol defined by the IETF and the ITU-T and is used in a distributed switching environment. It is designed as an internal protocol within a distributed system, which appears to the external world as a single VoIP gateway. Internally, the architecture is designed such that the intelligence of call control is outside of the gateways and handled by external agents (see Figure 2).

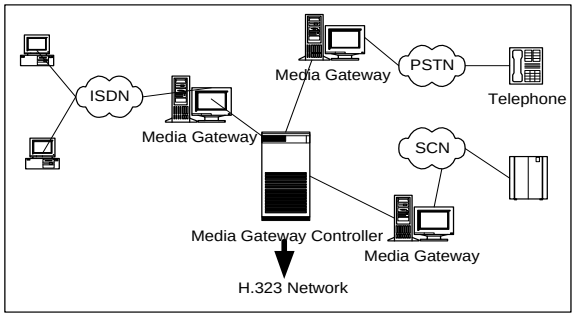

Figure 2. MEGACO Network Architecture.

Megaco thus divides the media logic and the signaling logic of a gateway across different functional components. While the Media Gateway (MG) handles the media logic part, the Media Gateway Controllers (MGCs, or Call Agents) control the Media Gateways to establish media paths through the distributed network. An MGC can control multiple MGs. On the other hand, one MG can register with multiple MGCs. Communication between these two functional units (MG and MGC) is governed by the Media Gateway Control Protocol (or

Megaco). Megaco is thus a master/slave protocol, where the call agents act as command initiators (or masters), and the MGs act as command responders (or slaves).

Session Initiation Protocol

Session Initiation Protocol (SIP), is a communication protocol used for a number of applications like Internet telephony, Call-forwarding, Multimedia conferencing, Terminal-type negotiation, Caller/callee authentication, and a host of other multimedia services. The standards for SIP [4] have been developed by the MMUSIC group of IETF.

SIP is transported typically over the connection-less UDP protocol. UDP is preferred over TCP because of its lower state-management overheads, real-time characteristics, and better performance. The standards for SIP include RFC2543 (Session Initiation Protocol), RFC2327 (Session Description Protocol) and a number of Internet-drafts, that are being worked upon. Figure 3 shows network components and sample message flows for SIP-based network.

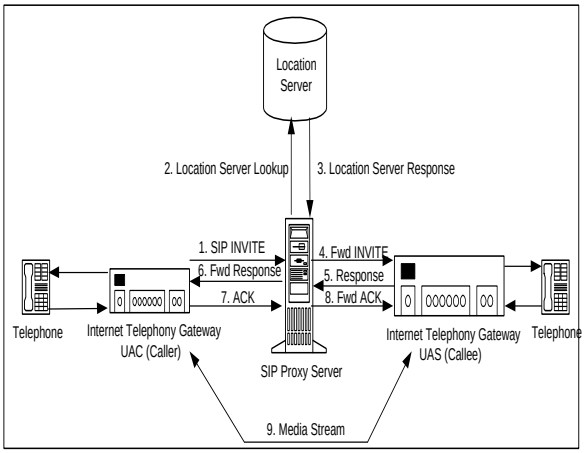

Figure 3. SIP Network components and message flows.

RTP and RTCP

RTP or Real Time Protocol provides support for applications with real-time properties, including timing reconstruction, loss detection, security and content identification. RTCP provides support for real-time conferencing for large groups within an Internet, including source-identification and support for gateways (like audio/video bridges), and multicast-to-unicast translators. RTP can be used without RTCP.

RTP is standardized in RFC1889. Besides this RFC, there are several other RFCs, which discuss specific problem areas of RTP.

3. Issues and Challenges

Quality of Service (QoS)

The biggest problem faced by voice over packet networks is that of providing the end users the Quality of Service (QoS) that they get in a traditional telephony network. Unlike the PSTN, where a dedicated end-to-end connection is established for a call, packet based networks use statistical multiplexing of the network resources. Though sharing resources amongst multiple users leads to a cost saving (and hence the attractiveness of the voice over packet phenomenon), it deteriorates the overall quality of service offered to a user. There are multiple parameters that determine the quality of service provided by a network. These include delay, the delay jitter and the packet loss experienced in a network. The next few subsections discuss these parameters in detail.

Delay

The Delay experienced in a packet network is classified into the following types: the Accumulation Delay, the Packetization Delay and the Network Delay. Each of these adds to the overall delay experienced by a user. The accumulation delay is caused by a need to collect a frame of voice samples for processing by the voice coder. This delay depends upon the sample time and the type of voice coder used. The accumulated voice samples are next encoded into a packet, which leads to the Packetization delay. Finally, once this packet is sent through the network, it experiences some delay to reach the destination. This is caused because of multiple factors, which includes the processing done by each intermediate node in the network to forward the voice packet, the capacity of the underlying physical medium, etc.

Delay in transporting a voice packet through the network leads to two main problems, namely Echo and Talker Overlap. Echo becomes a major problem when the round trip delay through the network becomes greater than 50 milliseconds. It is caused by the voice signal reflecting back from the end telephone equipment back to the speaker's telephone. To avoid the problems of echo in calls that have a round trip time of greater than 50 milliseconds, echo cancellers are required. Since packet networks introduce higher end-to-end delay, and hence have a greater round trip time, echo cancellation is an essential requirement for a voice over packet network. Apart from this, incase the end-to-end delay becomes greater than 250msec, the problem of Talker Overlap surfaces. This is experienced by one talker overlapping the other talker, and can be extremely annoying. Also, a delay of more than 250msec feels like a half-duplex connection and cannot be claimed to be an interactive session.

Jitter

In packet-based networks, two packets sent from the same source to the same destination might take different routes through the network. This is because the packets are routed through the network independently. Hence, two packets between the same source and destination might experience different processing delays and different congestion situations in the network, resulting in a variation in the overall delay experienced by the packets. This variation in the delay experienced by packets is measured as delay jitter. Also, this might lead to packets reaching the destination out of order. Unlike data packets, voice packets would be severely affected by the delay jitter.

To take care of the jitter, a buffering scheme is used at the destination. Packets at the destination are received and buffered. After the buffer is full to a threshold value, the packets are played in sequence and with a constant delay, minimizing the delay jitter. However, this buffering of packets

at the destination leads to an additional delay and also adds up to the other three types of delays discussed above.

Packet Loss

Since IP is a best effort protocol, packets routed through an IP network can be lost. To provide reliable transmission of data in an IP network, a retransmission scheme is used at the transport layer, which retransmits any packets for which an acknowledgement is not received from the destination (assuming that the packet got lost). However, the same scheme cannot be applied to voice, as a retransmitted voice packet might reach the destination much later than when it was expected. To compensate for lost packets, a few techniques are specified for voice over IP networks. Of these, one scheme proposes to play the last packet received again, to compensate for the lost packet. However, this scheme cannot work if there is a burst of lost packets. Another scheme proposes to send redundant information, increasing the bandwidth requirements. Thus, lost packet compensation is an essential requirement for supporting VoIP.

Billing and Interworking

Multiple variants exist today for supporting Voice over Packet Networks. Amongst these is Voice over ATM, Voice over IP and Voice over Frame Relay. All these variants would have to co-exist with each other, along with the conventional PSTN. This raises multiple issues, but primary amongst these is the issue of Interworking. Let us consider a scenario where a user in an IP network places a call to a user in a normal telephone network (PSTN). Then the call would have to traverse through a VoIP network to the PSTN through a gateway, which would have to provide the necessary Interworking, including address translation etc. Also, billing becomes complex in this scenario, since the call traverses networks that differ in principle. In a PSTN network, a dedicated connection is established between users, and they are charged even at times when there is silence and no voice samples are generated. However, packet based networks charge a user on the amount of bandwidth used, and not on the total duration of the call. Needless to say, error localization and reporting would also become more complex in such a heterogeneous scenario.

Transparency of Operation

As discussed above, the underlying mechanism supporting a voice call could be different, based upon the type of network (VoIP, VoATM, VoFR, PSTN, etc.) What is important in this scenario is to provide the user with a feel of a conventional telephone network. The call establishment process (call control and the associated signaling process) must be made transparent to the user. Also, a user placing a call from a PSTN telephone to a VoIP network, for example, should be unaware of the fact that his call is traversing two dissimilar networks.

APPLICATIONS

Internet telephony

Traditionally, Internet has been used for data applications. However, with improvements in packet-over-voice, Internet telephony has now become a hot area. Specifically, long-distance telephony has emerged as a killer application because it offers significant cost-savings vis-a-vis PSTN.

To provide long distance telephony, there are two possibilities. In the first model, the local access remains unchanged, while the IP network is used as a backbone between two voice switches/gateways. This model is beneficial in the sense that it does not mandate a PC with sound-card/microphone for telephony, thereby leaving the user interface unchanged while still providing cost benefits. The other model is to provide end-to-end IP connectivity, which requires additional efforts.

Internet-based call centers

VoIP opens exciting opportunities in call-center related activity. A user browsing through the product profile of a company may directly call online through a VoIP based call center. This facilitates better decision-making. Besides this, call-centers located at different places may be connected through a VoIP-based network.

Trunking Applications

Another VoIP application that holds promise is providing trunking facilities between head-office and branch-office using an IP link. The IP link can provide voice-only services, as well as integrated voice/date connectivity. The latter provides advantages that come along with network consolidation (e.g., reduced network management overheads, lower costs, etc.)

4. Fast Information Detection by using High Speed Time Delay Neural Networks

Finding certain information, in the incoming serial data, is a searching problem. First neural networks are trained to classify the required information from other examples and this is done in time domain. In information detection phase, each position in the incoming matrix is tested for presence or absence of the required information. At each position in the input one dimensional matrix, each sub-matrix is multiplied by a window of weights, which has the same size as the sub-matrix. The outputs of neurons in the hidden layer are multiplied by the weights of the output layer. When the final output is high, this means that the sub-matrix under test contains the required information and vice versa. Thus, we may conclude that this searching problem is a cross correlation between the incoming serial data and the weights of neurons in the hidden layer.

The convolution theorem in mathematical analysis says that a convolution of f with h is identical to the result of the following steps: let F and H be the results of the Fourier Transformation of f and h in the frequency domain. Multiply F and H* in the frequency domain point by point and then transform this product into the spatial domain via the inverse Fourier Transform. As a result, these cross correlations can be represented by a product in the frequency domain. Thus, by using cross correlation in the frequency domain, speed up in an

order of magnitude can be achieved during the detection process [5].

Assume that the size of the intrusion code is 1xn. In intrusion detection phase, a sub matrix I of size 1xn (sliding window) is extracted from the tested matrix, which has a size of 1xN. Such sub matrix, which may be an intrusion code, is fed to the neural network. Let W_i be the matrix of weights between the input sub-matrix and the hidden layer. This vector has a size of 1xn and can be represented as 1xn matrix. The output of hidden neurons h(i) can be calculated as follows:

$$h_i = g\left(\sum_{k=1}^{n} W_i(k) I(k) + b_i \right) \quad (1)$$

where g is the activation function and b(i) is the bias of each hidden neuron (i). Equation 1 represents the output of each hidden neuron for a particular sub-matrix I. It can be obtained to the whole input matrix Z as follows:

$$h_i(u) = g\left(\sum_{k=-n/2}^{n/2} W_i(k)\, Z(u+k) + b_i \right) \quad (2)$$

Eq.2 represents a cross correlation operation. Given any two functions f and d, their cross correlation can be obtained by:

$$d(x) \otimes f(x) = \left(\sum_{n=-\infty}^{\infty} f(x+n) d(n) \right) \quad (3)$$

Therefore, Eq. 2 may be written as follows [5]:

$$h_i = g\left(W_i \otimes Z + b_i \right) \quad (4)$$

where h_i is the output of the hidden neuron (i) and $h_i(u)$ is the activity of the hidden unit (i) when the sliding window is located at position (u) and (u) $\in$ [N-n+1].

Now, the above cross correlation can be expressed in terms of one dimensional Fast Fourier Transform as follows [5]:

$$W_i \otimes Z = F^{-1}\left(F(Z) \bullet F^{*}\left(W_i \right) \right) \quad (5)$$

Hence, by evaluating this cross correlation, a speed up ratio can be obtained comparable to conventional neural networks. Also, the final output of the neural network can be evaluated as follows:

$$O(u) = g\left(\sum_{i=1}^{q} W_o(i)\, h_i(u) + b_o \right) \quad (6)$$

where q is the number of neurons in the hidden layer. O(u) is the output of the neural network when the sliding window located at the position (u) in the input matrix Z. W_o is the weight matrix between hidden and output layer.

The complexity of cross correlation in the frequency domain can be analyzed as follows:

1- For a tested matrix of 1xN elements, the 1D-FFT requires a number equal to $N\log_2 N$ of complex computation steps [6]. Also, the same number of complex computation steps is required for computing the 1D-FFT of the weight matrix at each neuron in the hidden layer.

2- At each neuron in the hidden layer, the inverse 1D-FFT is computed. Therefore, q backward and (1+q) forward transforms have to be computed. Therefore, for a given matrix under test, the total number of operations required to compute the 1D-FFT is $(2q+1)N\log_2 N$.

3- The number of computation steps required by FTDNNs is complex and must be converted into a real version. It is known that, the one dimensional Fast Fourier Transform requires $(N/2)\log_2 N$ complex multiplications and $N\log_2 N$ complex additions [6]. Every complex multiplication is realized by six real floating point operations and every complex addition is implemented by two real floating point operations. Therefore, the total number of computation steps required to obtain the 1D-FFT of a 1xN matrix is:

$$\rho = 6((N/2)\log_2 N) + 2(N\log_2 N) \quad (7)$$

which may be simplified to:

$$\rho = 5N\log_2 N \quad (8)$$

4- Both the input and the weight matrices should be dot multiplied in the frequency domain. Thus, a number of complex computation steps equal to qN should be considered. This means 6qN real operations will be added to the number of computation steps required by FTDNNs.

5- In order to perform cross correlation in the frequency domain, the weight matrix must be extended to have the same size as the input matrix. So, a number of zeros = (N-n) must be added to the weight matrix. This requires a total real number of computation steps = q(N-n) for all neurons. Moreover, after computing the FFT for the weight matrix, the conjugate of this matrix must be obtained. As a result, a real number of computation steps = qN should be added in order to obtain the conjugate of the weight matrix for all neurons. Also, a number of real computation steps equal to N is required to create butterflies complex numbers ($e^{-jk(2\Pi n/N)}$), where 0<K<L. These (N/2) complex numbers are multiplied by the elements of the input matrix or by previous complex numbers during the computation of FFT. To create a complex number requires two real floating point operations. Thus, the total number of computation steps required for FTDNNs becomes:

$$\sigma = (2q+1)(5N\log_2 N) + 6qN + q(N-n) + qN + N \quad (9)$$

which can be reformulated as:

$$\sigma = (2q+1)(5N\log_2 N) + q(8N-n) + N \quad (10)$$

6- Using sliding window of size 1xn for the same matrix of 1xN pixels, q(2n-1)(N-n+1) computation steps are required when using CTDNNs for certain information detection or processing (n) input data. The theoretical speed up factor η can be evaluated as follows:

$$\eta = \frac{q(2n-1)(N-n+1)}{(2q+1)(5N\log_2 N) + q(8N-n) + N} \quad (11)$$

Time delay neural networks accept serial input data with fixed size (n). Therefore, the number of input neurons equals to (n). Instead of treating (n) inputs, the proposed new approach is to collect all the incoming data together in a long vector (for example 100xn). Then the input data is tested by time delay neural networks as a single pattern with length L (L=100xn). Such a test is performed in the frequency domain as described before. The combined information in the incoming data may have real or complex values in a form of one or two dimensional array. Complex-valued neural networks have many applications in fields dealing with complex numbers such as telecommunications, speech recognition and image processing with the Fourier Transform [7]. Complex-valued neural networks mean that the inputs, weights, thresholds and the activation function have complex values. In this section, formulas for the speed up ratio with different types of inputs (real /complex) will be presented. Also, the speed up ratio in case of a one and two dimensional incoming input matrix will be concluded. The operation of FTDNNs depends on computing the Fast Fourier Transform for both the input and weight matrices and obtaining the resulting two matrices. After performing dot multiplication for the resulting two matrices in the frequency domain, the Inverse Fast Fourier Transform is determined for the final matrix. Here, there is an excellent advantage with FTDNNs that should be mentioned. The Fast Fourier Transform is already dealing with complex numbers, so there is no change in the number of computation steps required for FTDNNs. Therefore, the speed up ratio in case of complex-valued time delay neural networks can be evaluated as follows:

1) In case of real inputs

A) For a one dimensional input matrix

Multiplication of (n) complex-valued weights by (n) real inputs requires (2n) real operations. This produces (n) real numbers and (n) imaginary numbers. The addition of these numbers requires (2n-2) real operations. The multiplication and addition operations are repeated (N-n+1) for all possible sub matrices in the incoming input matrix. In addition, all of these procedures are repeated at each neuron in the hidden layer. Therefore, the number of computation steps required by conventional neural networks can be calculated as:

$$\theta=2q(2n-1)(N-n+1) \tag{12}$$

The speed up ratio in this case can be computed as follows:

$$\eta=\frac{2q(2n-1)(N-n+1)}{(2q+1)(5N\log_2 N)+q(8N-n)+N} \tag{13}$$

The theoretical speed up ratio for searching short successive (n) code in a long input vector (L) using complex-valued time delay neural networks is shown in Tables 1, 2, and 3. Also, the practical speed up ratio for manipulating matrices of different sizes (L) and different sized weight matrices (n) using a 2.7 GHz processor and MATLAB is shown in Table 4.

B) For a two dimensional input matrix

Multiplication of (n^2) complex-valued weights by (n^2) real inputs requires ($2n^2$) real operations. This produces (n^2) real numbers and (n^2) imaginary numbers. The addition of these numbers requires ($2n^2$-2) real operations. The multiplication and addition operations are repeated $(N-n+1)^2$ for all possible sub matrices in the incoming input matrix. In addition, all of these procedures are repeated at each neuron in the hidden layer. Therefore, the number of computation steps required by conventional neural networks can be calculated as:

$$\theta=2q(2n^2-1)(N-n+1)^2 \tag{14}$$

The speed up ratio in this case can be computed as follows:

$$\eta=\frac{2q(2n^2-1)(N-n+1)^2}{(2q+1)(5N^2\log_2 N^2)+q(8N^2-n^2)+N} \tag{15}$$

The theoretical speed up ratio for detecting (nxn) real valued submatrix in a large real valued matrix (NxN) using complex-valued time delay neural networks is shown in Tables 5, 6, 7. Also, the practical speed up ratio for manipulating matrices of different sizes (NxN) and different sized code matrices (n) using a 2.7 GHz processor and MATLAB is shown in Table 8.

2) In case of complex inputs

A) For a one dimensional input matrix

Multiplication of (n) complex-valued weights by (n) complex inputs requires (6n) real operations. This produces (n) real numbers and (n) imaginary numbers. The addition of these numbers requires (2n-2) real operations. Therefore, the number of computation steps required by conventional neural networks can be calculated as:

$$\theta=2q(4n-1)(N-n+1) \tag{16}$$

The speed up ratio in this case can be computed as follows:

$$\eta=\frac{2q(4n-1)(N-n+1)}{(2q+1)(5N\log_2 N)+q(8N-n)+N} \tag{17}$$

The theoretical speed up ratio for searching short complex successive (n) code in a long complex-valued input vector (L) using complex-valued time delay neural networks is shown in Tables 9, 10, and 11. Also, the practical speed up ratio for manipulating matrices of different sizes (L) and different sized weight matrices (n) using a 2.7 GHz processor and MATLAB is shown in Table 12.

B) For a two dimensional input matrix

Multiplication of (n^2) complex-valued weights by (n^2) real inputs requires ($6n^2$) real operations. This produces (n^2) real numbers and (n^2) imaginary numbers. The addition of these numbers requires ($2n^2$-2) real operations. Therefore, the number of computation steps required by conventional neural networks can be calculated as:

$$\theta=2q(4n^2-1)(N-n+1)^2 \quad (18)$$

The speed up ratio in this case can be computed as follows:

$$\eta=\frac{2q(4n^2-1)(N-n+1)^2}{(2q+1)(5N^2\log_2 N^2)+q(8N^2-n^2)+N} \quad (19)$$

The theoretical speed up ratio for detecting (nxn) complex-valued submatrix in a large complex-valued matrix (NxN) using complex-valued neural networks is shown in Tables 13, 14, and 15. Also, the practical speed up ratio for manipulating matrices of different sizes (NxN) and different sized code matrices (n) using a 2.7 GHz processor and MATLAB is shown in Table 16.

An interesting point is that the memory capacity is reduced when using FTDNN. This is because the number of variables is reduced compared with CTDNN.

5. Conclusion

It goes without saying that packet-based voice has emerged as a cheap and attractive alternative to circuit-switched voice. The challenges and issues concerning VoIP notwithstanding, voice-networks based on IP will soon find a large market pie for themselves in the next generation mobile networks. To leverage faster deployment, ITU-T and IETF are developing standards to ensure vendor interoperability. In addition, the commercial deployment of VoIP based networks has already in great pace. Furthermore, new FTDNNs for fast detecting certain information in VOIP have been presented. Theoretical computations have shown that FTDNNs require fewer computation steps than conventional ones. This has been achieved by applying cross correlation in the frequency domain between the input data and the input weights of time delay neural networks. Simulation results have confirmed this proof by using MATLAB.

References

[1] U. Black, Voice over IP. Prentice Hall, 1999.

[2] D Rizzetto and C. Catania, "A Voice over IP Service Architecture for Integrated Communications," IEEE Internet Computing, pp. 53-62, 1999.

[3] Media Gateway Control Protocol (MGCP), 1999, www.ietf.org.

[4] H. Schulzrinne and J. Rosenberg, A Comparison of SIP and H.323 for Internet Telephony, 1998, www.cs.columbia.edu.

[5] H. M. El-Bakry, "New Faster Normalized Neural Networks for Sub-Matrix Detection using Cross Correlation in the Frequency Domain and Matrix Decomposition," Applied Soft Computing journal, vol. 8, issue 2, March 2008, pp. 1131-1149.

[6] J. W. Cooley, and J. W. Tukey, "An algorithm for the machine calculation of complex Fourier series," Math. Comput. 19, 297–301 (1965).

[7] R. Klette, and Zamperon, "Handbook of image processing operators, " John Wiley & Sonsltd, 1996.

[8] A. Hirose, "Complex-Valued Neural Networks Theories and Applications", Series on innovative Intellegence, vol.5. Nov. 2003.

[9] S. Jankowski, A. Lozowski, M. Zurada, " Complex-valued Multistate Neural Associative Memory," IEEE Trans. on Neural Networks, vol.7, 1996, pp.1491-1496.

Table 1: The theoretical speed up ratio for time delay neural networks (1D-real values input matrix, n=400).

Length of input matrix	Number of computation steps required for classical complex-valued neural networks	Number of computation steps required for fast complex-valued neural networks	Speed up ratio
10000	4.6027e+008	4.2926e+007	10.7226
40000	1.8985e+009	1.9614e+008	9.6793
90000	4.2955e+009	4.7344e+008	9.0729
160000	7.6513e+009	8.8219e+008	8.6731
250000	1.1966e+010	1.4275e+009	8.3823
360000	1.7239e+010	2.1134e+009	8.1571
490000	2.3471e+010	2.9430e+009	7.9752
640000	3.0662e+010	3.9192e+009	7.8237

Table 2: The theoretical speed up ratio for time delay neural networks (1D-real values input matrix, n=625).

Length of input matrix	Number of computation steps required for classical complex-valued neural networks	Number of computation steps required for fast complex-valued neural networks	Speed up ratio
10000	7.0263e+008	4.2919e+007	16.3713
40000	2.9508e+009	1.9613e+008	15.0452
90000	6.6978e+009	4.7343e+008	14.1474
160000	1.1944e+010	8.8218e+008	13.5388
250000	1.8688e+010	1.4275e+009	13.0915
360000	2.6932e+010	2.1134e+009	12.7433
490000	3.6674e+010	2.9430e+009	12.4612
640000	4.7915e+010	3.9192e+009	12.2257

Table 3: The theoretical speed up ratio for time delay neural networks (1D-real values input matrix, n=900).

Length of input matrix	Number of computation steps required for classical complex-valued neural networks	Number of computation steps required for fast complex-valued neural networks	Speed up ratio
10000	9.823 e+008	4.2911e+007	22.8933
40000	4.2206e+009	1.9612e+008	21.5200
90000	9.6176e+009	4.7343e+008	20.3149
160000	1.7173e+010	8.8217e+008	19.4671
250000	2.6888e+010	1.4275e+009	18.8356
360000	3.8761e+010	2.1134e+009	18.3409
490000	5.2794e+010	2.9430e+009	17.9385
640000	6.8985e+010	3.9192e+009	17.6018

Table 4: Practical speed up ratio for time delay neural networks (1D-real values input matrix).

Length of input matrix	Speed up ratio (n=400)	Speed up ratio (n=625)	Speed up ratio (n=900)
10000	17.88	25.94	35.21
40000	17.19	25.11	34.43
90000	16.65	24.56	33.59
160000	16.14	24.14	33.05
250000	15.89	23.76	32.60
360000	15.58	23.23	32.27
490000	15.28	22.87	31.99
640000	14.08	22.54	31.78

Table 5: The theoretical speed up ratio for time delay neural networks (2D-real values input matrix, n=20).

Size of input matrix	Number of computation steps required for classical complex-valued neural networks	Number of computation steps required for fast complex-valued neural networks	Speed up ratio
100x100	3.1453e+008	4.2916e+007	7.3291
200x200	1.5706e+009	1.9610e+008	8.0091
300x300	3.7854e+009	4.7335e+008	7.9970
400x400	6.9590e+009	8.8203e+008	7.8898
500x500	1.1091e+010	1.4273e+009	7.7711
600x600	1.6183e+010	2.1130e+009	7.6585
700x700	2.2233e+010	2.9426e+009	7.5556
800x800	2.9242e+010	3.9186e+009	7.4623

Table 6: The theoretical speed up ratio for time delay neural networks (2D-real values input matrix, n=25).

Size of input matrix	Number of computation steps required for classical complex-valued neural networks	Number of computation steps required for fast complex-valued neural networks	Speed up ratio
100x100	4.3285e+008	4.2909e+007	10.0877
200x200	2.3213e+009	1.9609e+008	11.8380
300x300	5.7086e+009	4.7334e+008	12.0602
400x400	1.0595e+010	8.8202e+008	12.0119
500x500	1.6980e+010	1.4273e+009	11.8966
600x600	2.4863e+010	2.1130e+009	11.7667
700x700	3.4246e+010	2.9425e+009	11.6381
800x800	4.5127e+010	3.9185e+009	11.5163

Table 7: The theoretical speed up ratio for time delay neural networks (2D-real values input matrix, n=30).

Size of input matrix	Number of computation steps required for classical complex-valued neural networks	Number of computation steps required for fast complex-valued neural networks	Speed up ratio
100x100	5.4413e+008	4.2901e+007	12.6834
200x200	3.1563e+009	1.9608e+008	16.0966
300x300	7.9272e+009	4.7334e+008	16.7476
400x400	1.4857e+010	8.8201e+008	16.8444
500x500	2.3946e+010	1.4273e+009	16.7773
600x600	3.5193e+010	2.1130e+009	16.6552
700x700	4.8599e+010	2.9425e+009	16.5160
800x800	6.4164e+010	3.9185e+009	16.3745

Table 8: Practical speed up ratio for time delay neural networks (2D-real values input matrix).

Size of input matrix	Speed up ratio (n=20)	Speed up ratio (n=25)	Speed up ratio (n=30)
100x100	17.19	22.32	31.74
200x200	17.61	22.89	32.55
300x300	16.54	23.66	33.71
400x400	15.98	22.95	34.53
500x500	15.62	22.49	33.32
600x600	15.16	22.07	32.58
700x700	14.87	21.83	32.16
800x800	14.64	21.61	31.77

Table 9: The theoretical speed up ratio for time delay neural networks (1D-complex values input matrix, n=400).

Length of input matrix	Number of computation steps required for classical complex-valued neural networks	Number of computation steps required for fast complex-valued neural networks	Speed up ratio
100x100	9.2111e+008	4.2926e+007	21.4586
200x200	3.7993e+009	1.9614e+008	19.3706
300x300	8.5963e+009	4.7344e+008	18.1571
400x400	1.5312e+010	8.8219e+008	17.3570
500x500	2.3947e+010	1.4275e+009	16.7750
600x600	3.4500e+010	2.1134e+009	16.3245
700x700	4.6972e+010	2.9430e+009	15.9604
800x800	3.9192e+009	6.1363e+010	15.6571

Table 10: The theoretical speed up ratio for time delay neural networks (1D-complex values input matrix, n=625).

Length of input matrix	Number of computation steps required for classical complex-valued neural networks	Number of computation steps required for fast complex-valued neural networks	Speed up ratio
100x100	1.4058e+009	4.2919e+007	32.7558
200x200	5.9040e+009	1.9613e+008	30.1025
300x300	1.3401e+010	4.7343e+008	28.3061
400x400	2.3897e+010	8.8218e+008	27.0883
500x500	3.7391e+010	1.4275e+009	26.1934
600x600	5.3885e+010	2.1134e+009	25.4969
700x700	7.3377e+010	2.9430e+009	24.9324
800x800	9.5868e+010	3.9192e+009	24.4612

Table 11: The theoretical speed up ratio for time delay neural networks (1D-complex values input matrix, n=900).

Length of input matrix	Number of computation steps required for classical complex-valued neural networks	Number of computation steps required for fast complex-valued neural networks	Speed up ratio
100x100	1.9653e+009	4.2911e+007	45.7993
200x200	8.4435e+009	1.9612e+008	43.0519
300x300	1.9240e+010	4.7343e+008	40.6410
400x400	3.4356e+010	8.8217e+008	38.9450
500x500	5.3791e+010	1.4275e+009	37.6817
600x600	7.7544e+010	2.1134e+009	36.6920
700x700	1.0562e+011	2.9430e+009	35.8870
800x800	1.3801e+011	3.9192e+009	35.2134

Table 12: Practical speed up ratio for time delay neural networks (1D-complex values input matrix).

Length of input matrix	Speed up ratio (n=400)	Speed up ratio (n=625)	Speed up ratio (n=900)
10000	37.90	53.58	70.71
40000	36.82	52.89	69.43
90000	36.34	52.47	68.69
160000	35.94	51.88	68.05
250000	35.69	51.36	67.56
360000	35.28	51.02	67.15
490000	34.97	50.78	66.86
640000	34.67	50.56	66.58

Table 13: The theoretical speed up ratio for time delay neural networks (2D-complex values input matrix, n=20).

Size of input matrix	Number of computation steps required for classical complex-valued neural networks	Number of computation steps required for fast complex-valued neural networks	Speed up ratio
100x100	6.2946e+008	4.2916e+007	14.6674
200x200	3.1431e+009	1.9610e+008	16.0281
300x300	7.5755e+009	4.7335e+008	16.0040
400x400	1.3927e+010	8.8203e+008	15.7894
500x500	2.2197e+010	1.4273e+009	15.5519
600x600	3.2386e+010	2.1130e+009	15.3266
700x700	4.4493e+010	2.9426e+009	15.1206
800x800	5.8520e+010	3.9186e+009	14.9340

Table 14: The theoretical speed up ratio for time delay neural networks (2D-complex values input matrix, n=25).

Size of input matrix	Number of computation steps required for classical complex-valued neural networks	Number of computation steps required for fast complex-valued neural networks	Speed up ratio
100x100	8.6605e+008	4.2909e+007	20.1836
200x200	4.6445e+009	1.9609e+008	23.6856
300x300	1.1422e+010	4.7334e+008	24.1301
400x400	2.1198e+010	8.8202e+008	24.0333
500x500	3.3973e+010	1.4273e+009	23.8028
600x600	4.9746e+010	2.1130e+009	23.5427
700x700	6.8519e+010	2.9425e+009	23.2856
800x800	9.0290e+010	3.9185e+009	23.0418

Table 15: The theoretical speed up ratio for time delay neural networks (2D-complex values input matrix, n=30).

Size of input matrix	Number of computation steps required for classical complex-valued neural networks	Number of computation steps required for fast complex-valued neural networks	Speed up ratio
100x100	1.0886e+009	4.2901e+007	25.3738
200x200	6.3143e+009	1.9608e+008	32.2021
300x300	1.5859e+010	4.7334e+008	33.5045
400x400	2.9722e+010	8.8201e+008	33.6981
500x500	4.7904e+010	1.4273e+009	33.5640
600x600	7.0405e+010	2.1130e+009	33.3197
700x700	9.7225e+010	2.9425e+009	33.0412
800x800	1.2836e+011	3.9185e+009	32.7581

Table 16: Practical speed up ratio for time delay neural networks (2D-complex values input matrix).

Size of input matrix	Speed up ratio (n=20)	Speed up ratio (n=25)	Speed up ratio (n=30)
100x100	38.33	46.99	62.88
200x200	39.17	47.79	63.77
300x300	38.44	48.86	64.83
400x400	37.92	47.23	65.99
500x500	37.32	46.89	64.89
600x600	36.96	46.48	64.01
700x700	36.67	46.08	63.31
800x800	36.38	45.78	62.64