electronics

MDPI

Article

A Hybrid Spoken Language Processing System for Smart Device Troubleshooting

Praveen Edward James [1], Hou Kit Mun [1,*] and Chockalingam Aravind Vaithilingam [2]

1 School of Engineering, Taylor's University, Taylor's University Lakeside Campus, Subang Jaya, Selangor 47500, Malaysia; edwardahs@gmail.com
2 HiRes Laboratory, School of Engineering, Taylor's University, Taylor's University Lakeside Campus, No. 1, Jalan Taylor's, Subang Jaya, Selangor 47500, Malaysia; chockalingamaravind.vaithilingam@taylors.edu.my
* Correspondence: HouKit.Mun@taylors.edu.my

Received: 6 May 2019; Accepted: 11 June 2019; Published: 16 June 2019

Abstract: The purpose of this work is to develop a spoken language processing system for smart device troubleshooting using human-machine interaction. This system combines a software Bidirectional Long Short Term Memory Cell (BLSTM)-based speech recognizer and a hardware LSTM-based language processor for Natural Language Processing (NLP) using the serial RS232 interface. Mel Frequency Cepstral Coefficient (MFCC)-based feature vectors from the speech signal are directly input into a BLSTM network. A dropout layer is added to the BLSTM layer to reduce over-fitting and improve robustness. The speech recognition component is a combination of an acoustic modeler, pronunciation dictionary, and a BLSTM network for generating query text, and executes in real time with an 81.5% Word Error Rate (WER) and average training time of 45 s. The language processor comprises a vectorizer, lookup dictionary, key encoder, Long Short Term Memory Cell (LSTM)-based training and prediction network, and dialogue manager, and transforms query intent to generate response text with a processing time of 0.59 s, 5% hardware utilization, and an F1 score of 95.2%. The proposed system has a 4.17% decrease in accuracy compared with existing systems. The existing systems use parallel processing and high-speed cache memories to perform additional training, which improves the accuracy. However, the performance of the language processor has a 36.7% decrease in processing time and 50% decrease in hardware utilization, making it suitable for troubleshooting smart devices.

Keywords: NLP; speech recognition; FPGA; LSTM; acoustic modeling; troubleshooting

1. Introduction

Manipulating speech signals to extract relevant information is known as speech processing [1]. This work integrates an optimized realization of speech recognition with Natural Language Processing (NLP) and a Text to Speech (TTS) system to perform Spoken Language Processing (SLP) using a hybrid software-hardware design approach. SLP involves three major tasks, namely translating speech to text (speech recognition), capturing the intent of the text, action determination using data processing techniques (NLP), and responding to users through voice (Speech Synthesis). Long Short Term Memory cell (LSTM), a class of Recurrent Neural Networks (RNN), is currently the state-of-the-art for continuous word speech recognition and NLP, due to its ability to process sequential data [2].

There are several LSTM-based speech recognition techniques available in the literature. For end-to-end speech recognition, speech spectrograms are chosen directly as the pre-processing scheme and processed by a deep bidirectional LSTM network with a novel Connectionist Temporal Classification (CTC) output layer [3]. However, CTC fails to model the dependence of output frames on previous output labels. In LSTM networks, the training time increases with additional LSTM layers and a technique utilizing a single LSTM layer with a Weighted Finite State Transducer (WFST)-based decoding approach was designed to mitigate the problem [4]. The decoding approach is fast, since it uses WFST and enables effective utilization of lexicons, but the performance is limited. RNN-T, a transducer-based RNN, is used to perform automatic speech recognition by breaking down the overall model into three sub-models, namely acoustic models, pronunciation dictionary, and language models [5]. To enhance the performance, Higher Order Recurrent Neural Network (HORNN), a variant of RNN, was used. This technique reduces the complexity of LSTM but uses several connections from previous time steps to eliminate vanishing long-term gradients [6].

In this work, word-based speech recognition was used in a Bidirectional LSTM to create a simpler and a reliable speech recognition model. Most NLP applications include an LSTM-based model for off-line Natural Language Understanding (NLU) using datasets [7], LSTM networks for word labelling tasks in spoken language understanding [8], and personal information, combined with natural language understanding to target smart communication devices, such as Personal Digital Assistants (PDA) [9]. Additionally, LSTM-based NLP models were also utilized for POS tagging, semantic parsing on off-line datasets, and to calculate the performance of machine translation by Natural Language Generation (NLG) [10,11].

In such scenarios, large sale LSTM models are computation and memory intensive. To overcome this limitation, a load balance aware pruning method is used to compress the model, along with a scheduler and a hardware architecture called an Efficient Speech Recognition Engine (ESE) [12]. However, the random nature of the pruning technique leads to unbalanced computation and irregular memory access. Hence, a structured compression technique that eliminates these irregularities with a block-circulant matrix and a comprehensive framework called C-LSTM was designed [13].

To further improve performance and energy efficiency, Alternating Direction Method of Multipliers (ADMM) are used to reduce block size and training time [14]. These reductions are achieved by determining the exact LSTM model and its implementation-based on factors such as processing element design optimization, quantization, and activation limited by quantization errors. There are several parameters that assess the performance of the spoken language processing system. They include accuracy and training time for speech recognition, processing time, hardware utilization and F1 score for NLP, and F1 score for the entire system. These values are determined to understand the relevance of this system in the current scenario.

The speech recognition component of the proposed system is implemented as a five-layered DNN with a sequential input layer, a BLSTM-based classification layer, a fully connected layered modeler for word-level modeling, a dropout layer, and a SoftMax output layer. The NLP part of the proposed system is implemented on an FPGA, as a four-layered DNN uses the one-pass learning strategy, using an input layer, LSTM-based learning layer, prediction layer, and an output layer. The text-to-speech component is designed by invoking Microsoft's Speech Application Programming Interface (SAPI) interface from MATLAB code.

The proposed spoken language processing system is constructed by integrating these three components using an RS232 serial interface. The rest of the paper is organized as follows. Section 1 introduces the topic in view of current ongoing research. Section 2 presents the system design and implementation. Section 3 gives the results and discussion, and Section 4 concludes the topic with an insight into the future.

2. Design and Implementation

This section presents the design of the spoken language processing system by adopting a bottom-up design methodology. The individual components are the speech recognizer, natural language processor, speech synthesizer, the integration of the software-based speech recognition component, and the hardware language processor component using the serial RS232 interface. The concept visualization of the proposed application is shown in Figure 1. An acoustic modeling server incorporates the proposed software speech recognition component to provide global access to any specific smart device connected through the Internet of Things (IOT) cloud platform. The query text generated from this server is received by the hardware language processor through the IOT cloud. On the other hand, the hardware language processor is embedded within the smart device as a co-processor.

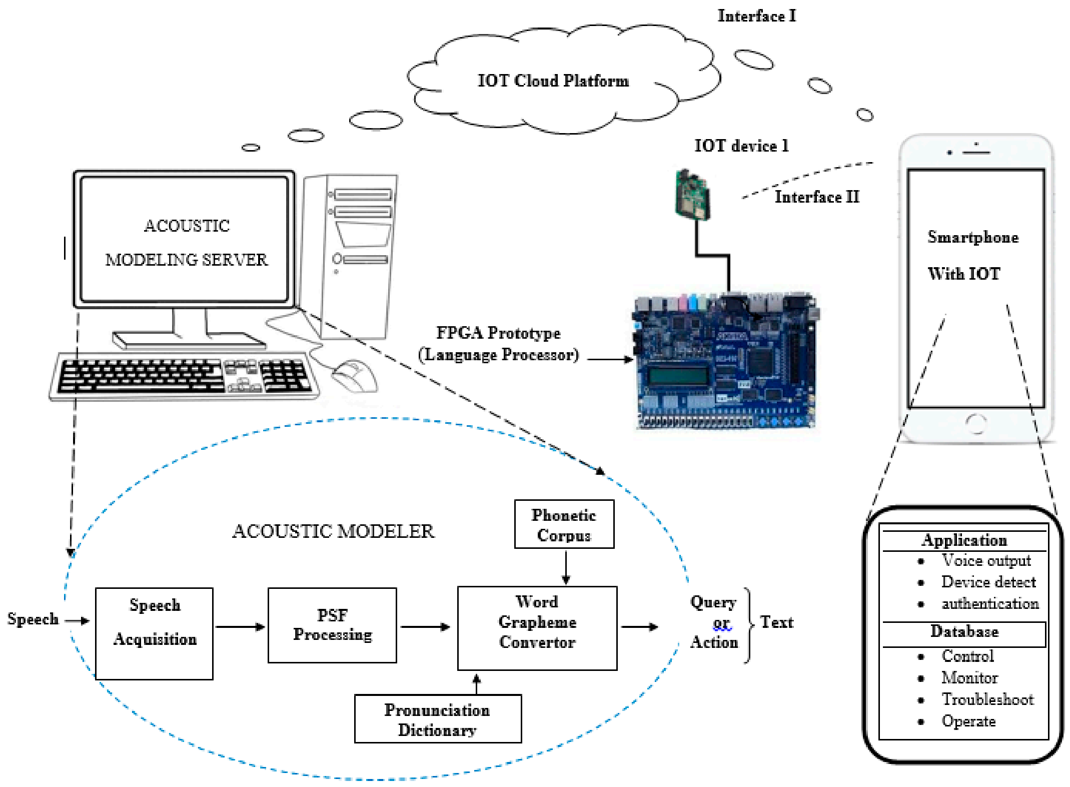


Figure 1. Smart device troubleshooting system.

The server communicates with the smart device through the IOT cloud and the smart device along with the language processor has an IOT device connected through an RS232 interface facilitating the communication process for purposes such as control, monitor, operate, and troubleshoot. The speech recognition component accepts speech in real-time and match features to words, which are combined as sentences. MATLAB R2018a is used to design this component as a software application.

The hardware-based language processor is coded using Verilog HDL, simulated using Modelsim 6.5 PE, synthesized, and verified experimentally. Finally, the entire system is tested with real-time data and its performance parameters are obtained. The design of the individual components is explained below.

2.1. BLSTM-Based Speech Recognition

Speech recognition is achieved by performing the following tasks. It involves capturing speech into the system using speech acquisition, pre-processing, feature extraction, BLSTM-based training and classification, word-level modeling, regularization with dropout, and likelihood estimation using SoftMax [15]. Speech acquisition is performed at the sentence-level using MATLAB-based recording software through a microphone. The speech acquisition process captures the speech signal from a natural environment, and hence it is susceptible to noise.

This degrades the signal strength and requires pre-processing the signal using several techniques. A band pass filter is initially used to filter signals between 50 Hz and 10,000 Hz. Pre-emphasis filter is a high-pass filter [16]. It is used to increase the amount of energy in the high frequencies and enables easy access to information in the higher formants of the speech signals. It enhances phone detection accuracy. The pre-emphasis filter function is given by Equation (1).

$$H(z) = 1 - az^{-1} \tag{1}$$

where a = 0.95.

The magnitude and phase response of the pre-emphasis filter is shown in Figure 2.

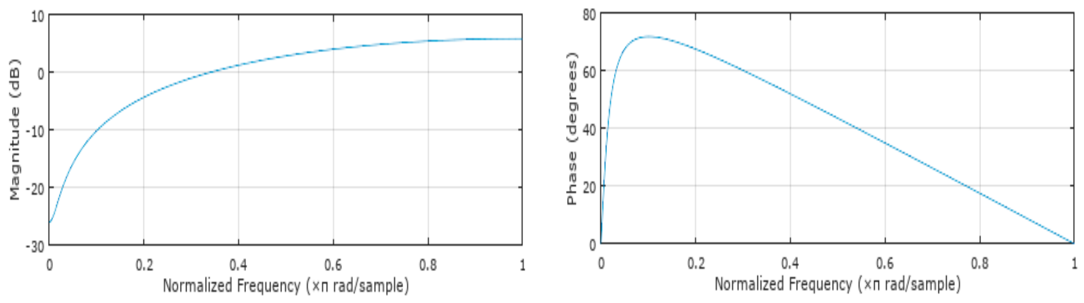


Figure 2. Magnitude and phase response of the pre-emphasis filter.

The speech signal is a natural realization of a set of randomly changing processes, where the underlying processes undergo slow changes. Speech is assumed to be stationary within a small section, called a frame. Frames are short segments of the speech signal that are isolated and processed individually. A frame contains discontinuities at its beginning and end. This can be smoothened by window functions. In this design, the Hamming window function is used and is given by Equation (2).

$$W(n) = \begin{Bmatrix} 0.54 - 0.46\ \cos\,(2\pi n/L) & \text{for } 0 \le n \le L \\ 0 & \text{otherwise} \end{Bmatrix} \tag{2}$$

where L is the length of the window. A good window function has a narrow main lobe and a low side lobe. A smooth tapering at the edges is desired to minimize discontinuities. Hence, the most common window used in speech processing is the Hamming window, which has lower side lobes, as the length of the window is increased [17]. The human ear is less sensitive to high frequencies. To overcome this problem, frequencies above 1000 Hz are mapped to Mel scale. Conversion to the Mel scale involves creating a bank of triangular filters that collects energy from each frequency band, with 10 filters spaced linearly within 1000 Hz and the rest spread logarithmically above 1000 Hz. The Mel Frequency Cepstral Coefficients (MFCC) can be computed from the raw acoustic frequency (f) using Equation (3).

$$mel(f) = 2595 * \log_{10}(1 + \frac{f}{700}) \tag{3}$$

Cepstrum is defined as the spectrum of the log of the spectrum of a time waveform [18]. The Cepstrum is used to improve phone recognition performance and a set of 12 coefficients is calculated. The Cepstrum of a signal is the Inverse DFT of the log magnitude of the Discrete Cosine Transform (DCT) of the signal. In addition to Mel Frequency Cepstral Coefficients (MFCC), the energy of the signal that correlates with the phone identity is computed. For a signal x in a window from time sample $t1$, to time sample $t2$, the energy of the n spectrum is calculated using Equation (4).

$$E(n) = \sum\nolimits_{t=t1}^{t2} x^2(t) \tag{4}$$

The energy forms the 13th coefficient and the set of 12 coefficients are combined to form 13 Cepstral coefficients. These Cepstral coefficients are used in a delta function (Δ) to identify changing features related to change in the spectrum [19]. For a short-time Cepstral sequence $C[n]$, the delta-Cepstral features are typically defined as

$$D[n] = C[n+m] - C[n-m] \tag{5}$$

where C refers to the existing coefficients, m is the number of coefficients used to compute Δ; m is set to 2. A set of 13 delta coefficients and then double delta coefficients (delta of delta coefficients) are calculated. By combining the three sets of 13 coefficients, the feature vectors with 39 coefficients are obtained. The LSTM model learns directly from the extracted features. However, they are labelled at the word level first. Labelling enables the features to be identified uniquely and converted to categorical targets. The categorized labels are directly mapped to the sentences so that the process of classifying data according to the targets can be learned by the BLSTM model, as shown in Figure 3.

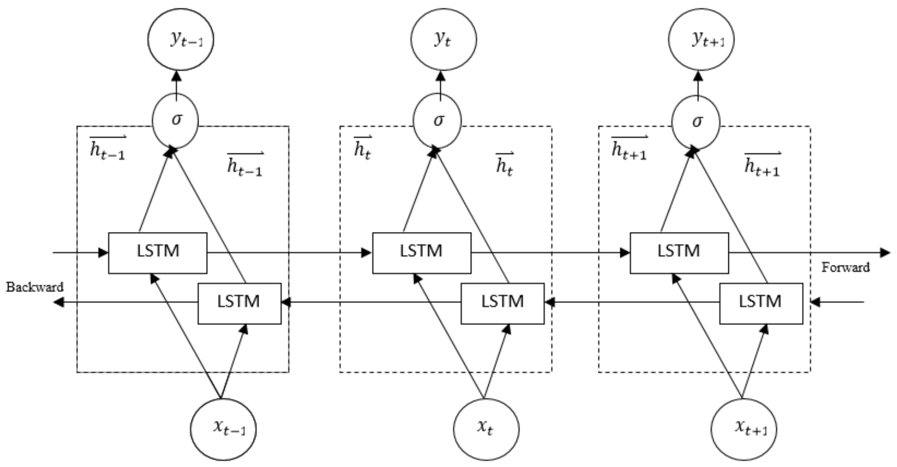


Figure 3. Bidirectional Long Short Term Memory Cell (BLSTM) structure.

The speech recognition component has multiple layers. The first layer is the input layer, followed by a BLSTM layer. This layer directly accepts two-dimensional feature vectors as inputs in a sequential manner, as the data varies with time. For training, the entire set of features representing a sentence is available. The BLSTM layer is trained to learn input patterns along with the categorized labels using the Back Propagation Through Time (BPTT) algorithm. The BPTT algorithm executes the following steps to train the BLSTM model.

Step 1: Feed-forward computation. Feed-forward computation is similar to a Deep Neural Network (DNN). Additionally, it has a memory cell in which past and future information can be stored. The cell is constrained by input, output, update, and forget gates to prevent random access.

Step 2: Gradient calculation. This is the mini-stochastic gradient calculation by taking the derivation of the cost function. The cost function measures the deviation of the actual output from the target function. In this study, cross-entropy is used to calculate the loss.

Step 3: Weight modification. The weights between the output layer and the input layer are updated with a combination of gradient values, learning rate, and a function of the output.

Step 4: Dropout. The process is repeated until the cost function is lesser than a threshold value. The training of the BLSTM is followed by regularization using an existing technique called dropout by adding a dropout layer. Dropout erases information from units randomly. This forces the BLSTM to compute lost information from the remaining data. It has a very positive effect on BLSTM, which stores past and future information in memory cells. An illustration of possible areas for application of Dropout is indicated with dotted lines in Figure 4. Dropout is applied only to the non-recurrent connections. This is indicated by dotted lines. Using dropout on these connections increases the robustness of the model and also minimizes over-fitting.

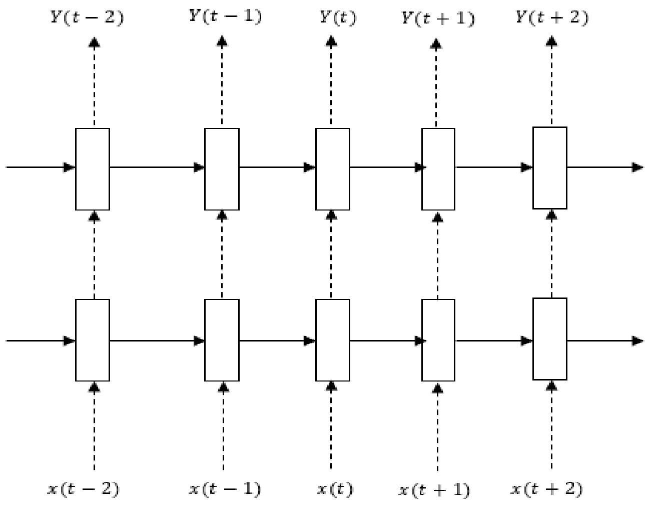


Figure 4. Dropout-based regularization.

where x represents the input, rectangular boxes indicate LSTM cells, and y represents the output. Word-level modeling is done using the TIMIT which is an acoustic corpus for speech data [20].

2.2. LSTM-Based Language Processor

The LSTM-based NLP framework is implemented as a hardware language processor. The language processor is designed using a combination of two implementation styles, namely process-based and model-based, and hence is hybrid in nature. The language processor is designed as a combination of functional blocks using Verilog Hardware Description Language (VHDL). Logical verification is done using Modelsim 6.5 PE. The design is prototyped on an Altera DE2-115 Board with CYCLONE IVE EP4CE115FC8 FPGA using Quartus II software. The various functional blocks are individually explained below.

2.2.1. Universal Asynchronous Receiver Transmitter (UART) Module

The UART module has a transmitter and a receiver to enable two-way communication with the software modules. It follows asynchronous communication controlled by the baud rate. The commencement and completion of transmission of a byte are indicated with start and stop bits. The receiver receives data from the speech recognition module one bit at a time. The receiver monitors the line for the start bit, which is a transmission from high to low. If the start bit is detected, the line is sampled for the 1st bit immediately after the start bit.

Once the 8 data bits are received, the data bits are processed into the system after the reception of a stop bit, which is a transition from low to high. The FPGA clock frequency is 50 MHz and for the proposed system, the transmission baud rate chosen is 9600 bps. At the receiver, when the data are sampled at the FPGA clock frequency, data cannot be retrieved. Hence, the data transmission rate of the transmitter and the reception rate at the receiver are kept constant and equal by dividing the clock frequency with the integer 5208 to generate the baud rate of 9600 kbps.

2.2.2. Vectorizer

The actual input to the language processor is text. However, the language processor handles only numeric data internally. Hence, the vectorizer converts the text to a byte value. The process of receiving query data from the input device is accomplished via a UART interface (receiver), shift register, and First-in-First-out (FIFO) register. The process flow diagram real-time processing of query text is shown in Figure 5. Each word of a query text is converted to a key in a sequence of steps:

1. The UART receiver receives a byte of data and transfers it to the shift register.

2. In the shift register, the maximum word length is fixed at 80 bits. It is monitored by a counter. The shift register receives data until the maximum length is reached or the word end is detected.
3. The FIFO length is variable. However, for the current implementation, it is fixed as the first 8 words of a sentence. At the end of each word, the FIFO moves on to the next value, when indicated by a counter's (~W/R) active-low signal. When the FIFO becomes full, the (~W/R) becomes active-high, indicating that the FIFO can be read sequentially.
4. The vectorizer module reads each word from the FIFO and converts it to an 8-bit value mapping the information dictionary, followed by a key encoder.
5. The Key encoder receives each byte value and encodes it into a unique 8-bit key.

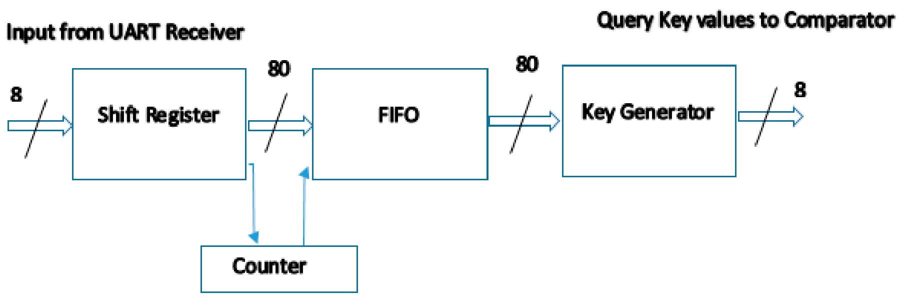
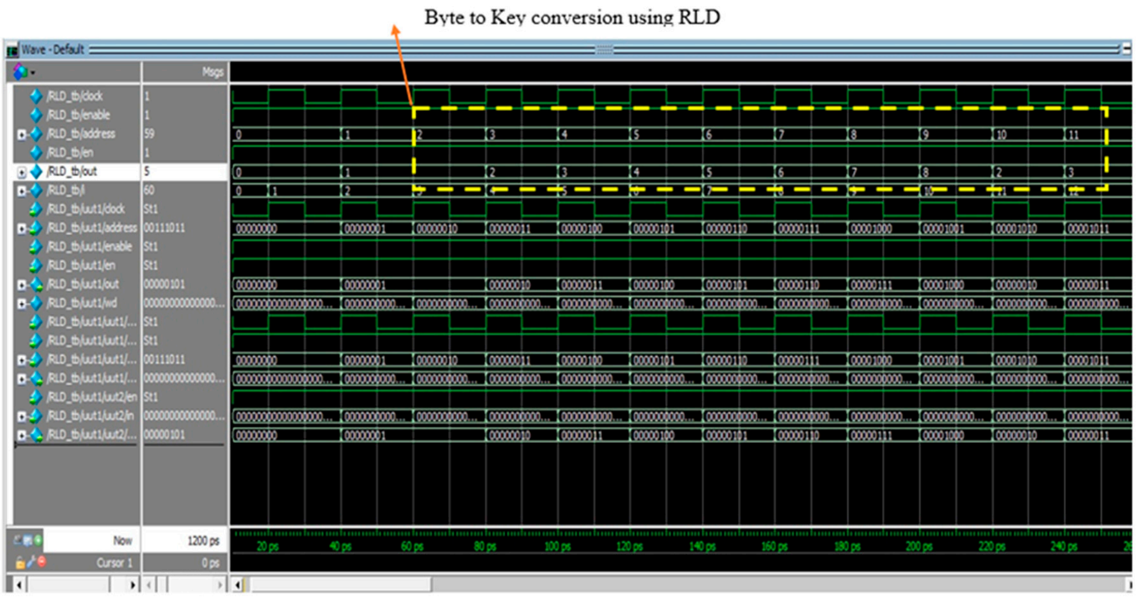


Figure 5. Vectorizer module.

The generation of the query text is followed by the generation of the information text.

2.2.3. Information Text Generation

The logic verification of the Reverse Lookup Dictionary (RLD) is given in Figure 6. Here, the number and its corresponding key are highlighted.



```
/RLD_tb/clock                1
/RLD_tb/enable               1
/RLD_tb/address              59
/RLD_tb/en                   1
/RLD_tb/out                  5
/RLD_tb/i                    60
/RLD_tb/uut1/clock           St1
/RLD_tb/uut1/address         00111011
/RLD_tb/uut1/enable          St1
/RLD_tb/uut1/en              St1
/RLD_tb/uut1/out             00000101
/RLD_tb/uut1/wd              0000000000000...
/RLD_tb/uut1/uut1/...        St1
/RLD_tb/uut1/uut1/...        St1
/RLD_tb/uut1/uut1/...        00111011
/RLD_tb/uut1/uut1/...        0000000000000...
/RLD_tb/uut1/uut2/en         St1
/RLD_tb/uut1/uut2/in         0000000000000...
/RLD_tb/uut1/uut2/...        00000101
```

Figure 6. Logic verification of reverse lookup dictionary (RLD) module.

A paragraph of information text is stored in a text file and downloaded into a custom RAM called Word Library during runtime. The location address of each word is 8-bit and corresponds to the location of the word library starting from 00 to FF. There is an address generator module for generating these addresses. Initially, the words are assigned byte values by the vectorizer. Then, unique keys are assigned for each byte value by the key generator and the combination of these blocks constitutes a Reverse Lookup Dictionary (RLD). RLD performs two different functions. It is used to generate a unique key from the word generation. It is also used to recover a word given the key.

2.2.4. Key Encoder

The information text keys and query text keys generated by the previous modules are compared by a comparator and assigned four 8-bit values based on four different observed criteria, as given in Table 1. From the table, it can be seen that if the start or end of the query text or information text is detected, unique values are assigned. The same process is performed when the match or no match of the two text sequences are detected. This process primarily standardizes the input sequence for one-pass learning and additionally secures the data from the external environment. The block diagram of the key encoder is shown in Figure 7.

Table 1. Key encoder.

No.	Comparison Parameters	Encoder Output
1	Qtext start	Output Value: 8′d127
2	Qtext end	Output Value: 8′d63
3	Itext == Qtext	Output Value: 8′d191
4	Itext <> Qtext	Output Value: 8′d0

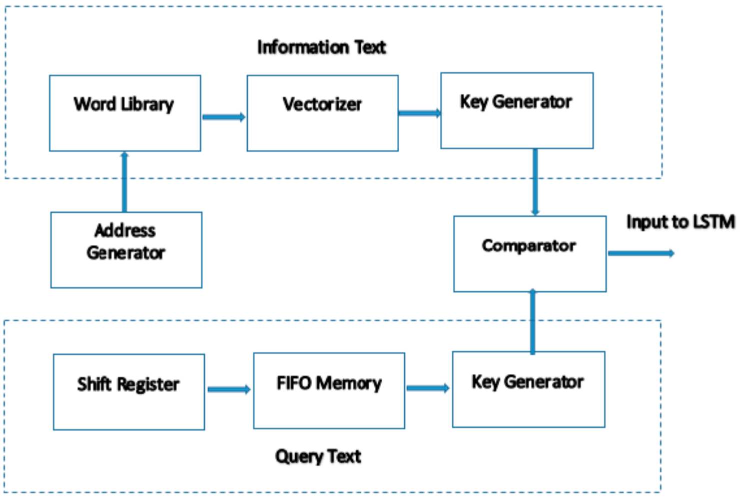


Figure 7. Process flow diagram of the key encoder.

2.2.5. LSTM Decoder

The values generated by the key encoder are allowed through the LSTM decoder. The LSTM decoder is a Deep Learning Network (DNN) consisting of an input layer, two LSTM hidden layers—namely the training and prediction layer—and an output layer. The LSTM layers train on the vales from the key encoder and predict the response by implementing self-learning within their structure using individual LSTM modules.

A LSTM layer has a memory cell of an LSTM module controlled by four gates, namely the input gate, output gate, update gate, and forget gate. The input gate allows new information to flow into the memory cell. The update gate allows information changes to be updated in the memory cell. The forget gate allows information in the memory cell to be erased. The output gate allows stored information to be transferred to subsequent modules or layers in the LSTM module.

The LSTM module is combined with weight and bias storage memory units to form the LSTM decoder. The key encoder values are sequentially transferred into the LSTM training layer by the input layer. In this layer, the four values generated by the encoder are random and are mapped to four ranges of values by the LSTM training layer. At the same time, the prediction layer keeps track of the order of the sequence of values and predicts the index in the word library, where most of the values are located. The RLD is used as reference for this task. If multiple indices are obtained in this process, identification of the most probable information text that constitutes the response is achieved by the dialogue modeler. The block diagram of the LSTM decoder is shown in Figure 8.

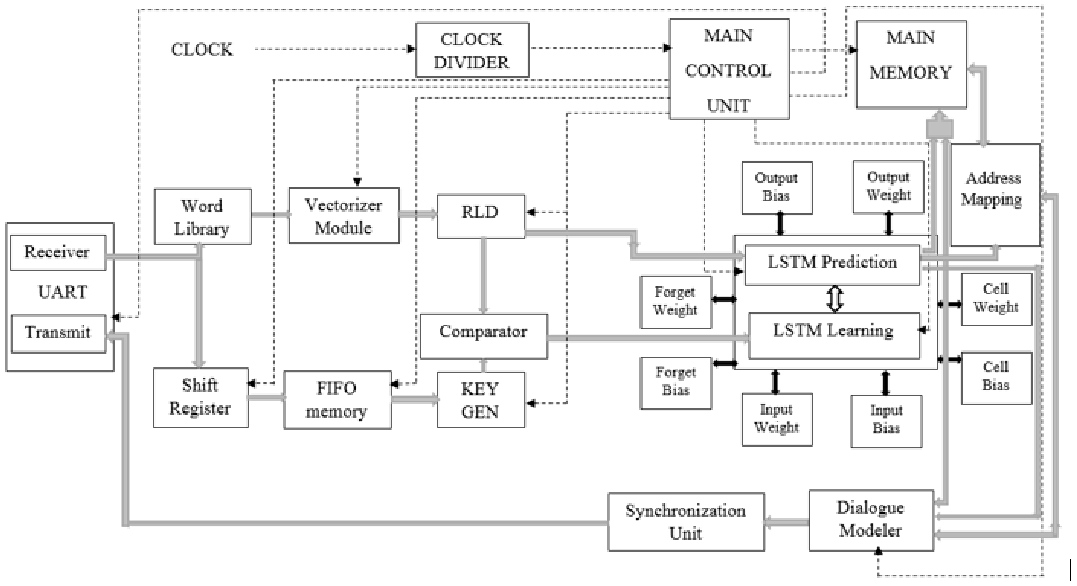


Figure 8. Long Short Term Memory Cell (LSTM) decoder.

2.2.6. Dialogue Modeler

The dialogue modeler consists of index detect, output hold, and output generation modules. The index detect compares the indices of information text generated in comparison with query text and calculates a score based on the matching keywords and the sequence in which these words occur. The language processor is a synchronous module and generates the output in a sequence. Hence, the output is maintained at the previous value until the similarity score is calculated by the index detection module. This operation is performed by the output hold module. Finally, the byte values of the actual words in a sentence are generated by the output generation module. The entire processor operates using a system clock and two control signals generated by the main control unit.

2.2.7. Main Control Unit

This is used to synchronize all the individual hardware units. Most of the hardware blocks execute sequentially and are controlled using enable signals. The state diagram is given in Figure 9. The UART receiver operates at the baud rate, hence the shift register needs to operate with a clock slower than the on-board clock. Hence, a clock divider circuit is used to reduce the clock speed. This circuit is under the control of the main control unit. A shift signal is generated to shift each byte of input data into the shift register. The main control unit is implemented as a state machine. The FIFO is operated in write or read mode using a (write/read) signal.

The control block enables the query data captured by the FIFO from the shift register. This data is transferred to the subsequent modules by generating a query enable signal (QEN) after the start signal (START) is received. The query text is simultaneously transferred to the subsequent modules by transferring the query text until all input words are captured in the FIFO. Then, the simultaneous transfer of query text and information text is achieved by enabling an information-text-enable signal (IEN). If a done signal (DONE) is received the state machine returns to the initial state.

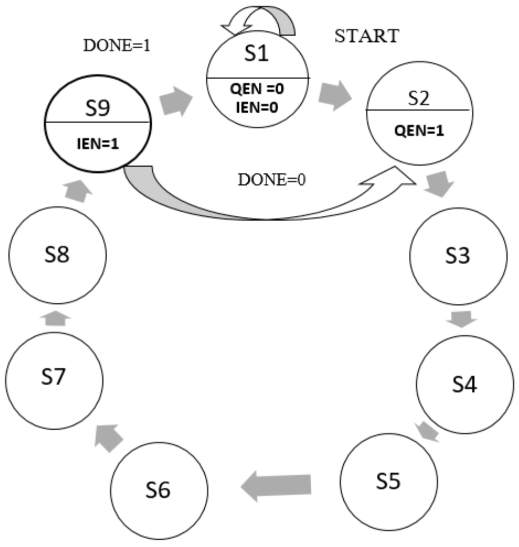


Figure 9. Main control unit of the language processor.

The weights are predetermined through a series of simulations such that the values of the cell lie in the range of 25% and 75% of the maximum possible value. The learning algorithm is implemented in the structure of the LSTM network and is explained below:

Step 1: In the first step, the query text is vectorized and assigned unique keys.

Step 2: The information text is also vectorized, assigned unique keys, and populated in a look-up dictionary.

Step 3: The key encoder sequentially compares every query key with the information key, encodes them into 4 possible values, and transfers them sequentially to the LSTM network.

Step 4: The encoded keys are mapped to four ranges of values and the index of the response text is identified.

Step 5: The integer values of the response text following the index are extracted from the reverse lookup dictionary to obtain response information sentences.

Step 6: Finally, a proximity score is calculated based on the similarity between information and query sentence by the dialogue modeler.

Step 7: The sentence with the maximum score is selected as the most probable response sentence.

The experimental setup of the spoken language processing system is shown in Figure 10.

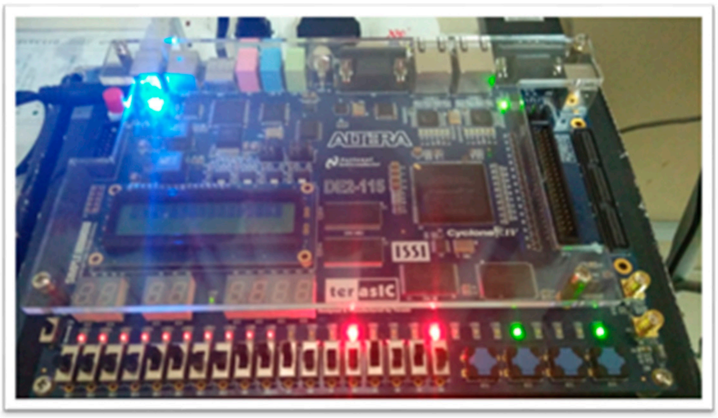


Figure 10. Response text from the language processor.

(A Demo of the Spoken Language processing system is presented in the YouTube link below: https://www.youtube.com/watch?v=T_6ntzVBRWI).

3. Results and Discussion

3.1. Experiment 1

The spoken language processing system is tested by performing three sets of experiments. In experiment 1, the BLSTM-based speech recognition component is tested. The dataset consists of 100 sentences from a vocabulary of 245 words. The proposed BLSTM-based speech recognition system (BLSTM2) was trained on the 100 sentences used earlier to evaluate the performance of the HMM speech recognition system. To overcome the possibilities of over-fitting, dropout-based regularization was used. The effect of dropout in the model's accuracy was evaluated by testing with and without the addition of a dropout layer. During prediction, the real-time input was parameterized and the text sentence for the speech signals was generated by comparing with trained parameters. The results are shown in Table 2.

Table 2. Accuracy of BLSTM system.

No.	Sentences Used	Accuracy (%)	
		BLSTM 1	**BLSTM 2**
1	This is Samsung S6	90	90
2	Lee Byung is Samsung founder	90	90
3	S6 has a touch screen	100	100
4	Is camera working?	80	90
5	Samsung has Android OS	80	80
6	The Wi-Fi is ON	80	70
7	My name is Max	50	70
8	What is your name?	70	70
9	Hi to all!	100	80
10	Conclude the event	70	80
	Average Accuracy	81	82

From Table 2, the accuracy of the BLSTM-based speech recognition system was obtained as 82%. To compare and evaluate the performance, a custom designed HMM-based continuous word speech recognition system was implemented. The accuracy of the HMM-based system was obtained from Table 3 as 77.03%.

Table 3. Word Error Rate of the three Hidden Markov Model (HMM)-based models.

No.	Sentences Used	HMM 1	HMM 2	HMM 3
1	This is Samsung S6	26.1	25.6	25.8
2	Lee Byung is Samsung founder	20.8	21.1	21.7
3	S6 has a touch screen.	19.4	20.3	17.2
4	Is camera working?	24.5	23.3	25.4
5	Samsung has Android OS	26.1	24.8	24.5
6	The Wi-Fi is ON	24.1	25.2	21.5
7	My name is Max	24.6	25.5	24.1
8	What is your name?	25.1	24.9	23.9
9	Hi to all!	23.9	22.1	24.3
10	Conclude the event.	23.3	20.3	21.3
Average Word Error Rate (WER)		23.79	23.31	22.97

These results indicate the improvement in accuracy by implementing the BLSTM-based speech recognition system.

3.2. Experiment 2

In this experiment, the second component of the spoken language processing system, namely the language processor, is tested. This system is LSTM-based and is implemented on an Altera DE2-115 board as dedicated hardware. The input is processed as an encoded sequence of hexadecimal bytes for every character of a sentence and is decoded as the response text by the LSTM network. The performance metrics include processing time and F1 score. The output of the LSTM layer is several possible responses and the best possible response is selected by a scoring mechanism using a dialogue modeler. The results are shown in Table 4.

Table 4. F1 score of language processor.

No.	Query and Response	Accuracy (%)	Processing Time (s)	F1 Score
1	Give the founder of Samsung? The founder of Samsung is Lee Byung	90	0.59	94.7
2	Identify the current focus of Samsung Samsung current focus is on smartphones	90	0.59	94.7
3	Which OS is present in S6? S6 uses Android OS	90	0.59	94.7
4	Give the location of Samsung? Samsung location is in South Korea	90	0.59	94.7
5	S6 was released in? S6 was released in 2015	90	0.59	94.7
6	What is the main feature of S6 S6 main feature is cost-effective	90	0.59	94.7
7	Tell the RAM size of S6 S6 RAM size is 3GB	90	0.59	94.7
8	How much does S6 cost? S6 cost is 1000 Ringgits	90	0.59	94.7
9	Is Samsung a local company? Samsung is an Multinational (MNC) company	90	0.59	94.7
10	The main competitor of Samsung is Samsung main competitor is Apple	100	0.59	100
	AVERAGE	91	0.59	95.2

The FPGA device used in the design is the Cyclone IVE EP4CE115F29C8. The Computer Aided Design (CAD) tool used for the design is Altera Quartus II version 10.1. The logic cells used are 4617, among which 4475 cells are used for combinational functions and 490 cells are used for registered processing (memory-based). An alternate system, called a process-based NLP system, is designed for performance comparison. The process-based NLP system uses syntactic and semantic rules, which extract response information from the query text and generates the response text.

This system uses the same dataset of 100 sentences used in the language processor. The results are presented in Table 5. The results reveal that the performance of the language processor is better in terms of the F1 score and processing time. The language processor has an improvement of 0.7% in F1 score, with a 2.6× times decrease in processing time over the process-based system.

Table 5. F1 score of process-based system.

No.	Query and Response	Accuracy (%)	Processing Time (s)	F1 Score (%)
1	Give the founder of Samsung? The founder of Samsung is Lee Byung.	100	2.12	94.7
2	Identify the current focus of Samsung Samsung current focus is on smartphones	90	2.11	88.8
3	Which OS is present in S6? S6 uses Android OS	100	2.11	100
4	Give the location of Samsung? Samsung location is in South Korea	100	2.11	100
5	S6 was released in? S6 was released in 2015	90	2.13	88.8
6	What is the main feature of S6 S6 main feature is cost-effective	90	2.11	94.7
7	Tell the RAM size of S6 S6 RAM size is 3GB	90	2.13	94.7
8	How much does S6 cost? S6 cost is 1000 Ringgits	90	2.13	94.7
9	Is Samsung a local company Samsung is an MNC company	90	2.12	94.7
10	The main competitor of Samsung is Samsung main competitor is Apple	90	2.12	94.7
	AVERAGE	93	2.12	94.58

3.3. Experiment 3

In this experiment, the HMM-based speech recognition system (HMM3) and BLSTM-based speech recognition system (BLSTM2) are combined with each of the NLP systems, namely the process-based NLP system (NLP1) and language processor (NLP2), to form four Spoken Language Processing Systems (SLPS). HMM3 and NLP1 systems are combined to form SLPS1. HMM3 and NLP2 systems are combined to form the SLPS2 system. Similarly, BLSTM and NLP1 are combined to form the SLPS3 system and BLSTM and NLP2 are combined to form SLPS4.

Each system is trained and tested based on 20 query sentences within a vocabulary of 243 words to constitute a training dataset of 200 sentences. The performance of these systems are accessed with their F1 scores and the results are shown in Table 6. The F1 scores of the four systems SLPS1, SLPS2, SLPS3, and SLPS4 are obtained from Table 6 as 81.3, 84.7, 84.05, and 88.2, respectively. The hybrid speech processing system has a recognition accuracy of 81.5%, hardware utilization of 5%, processing time of 0.59 s, and F1 score of 88.2%.

The hardware utilization of the language processor is shown in Figure 11.

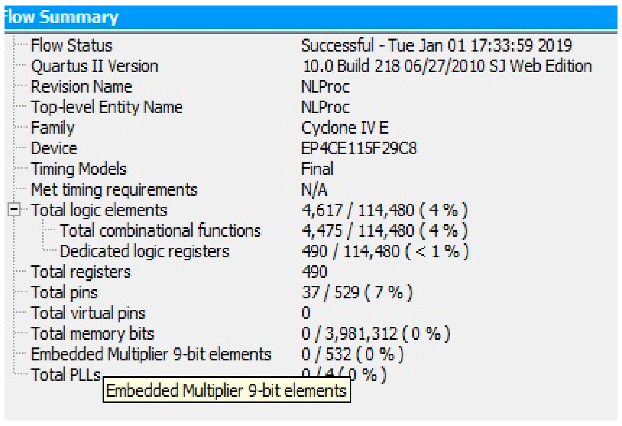

```
low Summary
Flow Status                          Successful - Tue Jan 01 17:33:59 2019
Quartus II Version                   10.0 Build 218 06/27/2010 SJ Web Edition
Revision Name                        NLProc
Top-level Entity Name                NLProc
Family                               Cyclone IV E
Device                               EP4CE115F29C8
Timing Models                        Final
Met timing requirements              N/A
Total logic elements                 4,617 / 114,480 ( 4 % )
    Total combinational functions    4,475 / 114,480 ( 4 % )
    Dedicated logic registers        490 / 114,480 ( < 1 % )
Total registers                      490
Total pins                           37 / 529 ( 7 % )
Total virtual pins                   0
Total memory bits                    0 / 3,981,312 ( 0 % )
Embedded Multiplier 9-bit elements   0 / 532 ( 0 % )
Total PLLs                           0 / 4 ( 0 % )
```

Figure 11. Hardware utilization of language processor.

Table 6. F1 scores of various Spoken Language Processing System (SLPS) implementations.

No.	Query Sentence	SLPS 1	SLPS 2	SLPS 3	SLPS 4
1	Where is Samsung located	40	70	40	70
2	Who is the founder of Samsung	80	80	60	70
3	What is the current focus of Samsung	80	70	80	90
4	When was S6 released	70	100	50	50
5	Which OS is present in Samsung	90	80	80	100
6	Is Samsung a national company	50	80	70	70
7	The main Competitor of Samsung is	90	60	80	70
8	How much does S6 cost	40	70	90	80
9	Tell the RAM size of S6	50	40	50	70
10	Airasias main office is located in	60	60	80	100
11	AK021′s arrival time is	50	70	90	100
12	Alternate flight to Perth	40	60	70	70
13	Research office is at	100	90	80	70
14	Does the computer have a name	50	60	90	90
15	Coming Eureca conference happens in	70	80	70	60
16	Conclude the Event	80	70	70	90
17	Greet everyone	70	80	80	70
18	Your native is	70	70	60	80
19	Which university are you studying at	90	100	80	100
20	Thank you for the details	100	80	80	80
	F1 Score (%)	81.30	84.7	84.05	88.2
	Standard deviation	2.00	1.42	1.41	1.44

The performance metrics of several existing systems are compared with the existing system and are shown in Table 7.

Table 7. Result Analysis.

No.	System Type (Real-Time)	Speech Recognition		Natural Language Processing			References
		Accuracy (%)	Training Time (s/Epoch)	Processing Time (s)	Hardware Utilization (%)	F1 Score (%)	
1	Full Software (V)	85.05	1.1			95.6	[8]
2	Full Hardware (X)	79.31		0.082 ms	74.7		[14]
3	Full Hardware (Y)			0.932	10.1	89.1	[21]
4	Full Hardware (Z)	−79.68		0.932	61.3		[13]
5	Proposed System (Software/Hardware)	81.5	1.125	0.59	5	88.2	
	Variation with Existing System	<4.15%	>2.27%	<36.7%	<50.4%	<7.7%	

The performance of the hardware-based NLP is tested using processing time, hardware utilization, and F1 score. In a speech recognition task, the proposed system has a 4.15% decrease in recognition accuracy compared with System V [8]. Similarly, the proposed system has a 2.27% increase in training time compared with System V [8]. The increase in recognition accuracy and marginal training time decrease from the proposed system is due to the inherent parallelism and cache mechanisms used in the system.

In a hardware-based NLP task, the proposed system has a 36.7% decrease in processing time, a 50.4% decrease in hardware utilization, and a 7.7% decrease in F1 score in comparison with System y [21]. The result analysis of the proposed system is shown in Table 7. The decrease in F1 score is due to the reduced accuracy of the speech recognition task, as the query text is the output of the speech recognizer. This shows that the decrease in processing time and hardware utilization are significant.

4. Conclusions and Future Work

The final implementation of the spoken language processing system had a recognition accuracy of 81.5%, 45 s training time for speech recognition, 0.59 s of processing time, 5% hardware utilization, and F1 scores of 95.2% for the language processor and 88.2% for the whole hybrid system. The training time limitation of the LSTM components is overcome in the proposed system using a hybrid speech processing system. The hybrid system incorporates an FPGA-based language processor with a unique architecture and one-pass learning. The language processor is more suited for Application Specific Integrated Circuit (ASIC) implementation as a coprocessor in low-cost electronic devices. The ongoing process is the full-fledged application of the proposed system for the task of smart device troubleshooting.

Author Contributions: Conceptualization and Writing, P.E.J. and H.K.M.; Supervision, Review and editing of the manuscript, H.K.M. and C.A.V.

Funding: This research was funded by Taylor's University.

Acknowledgments: The authors would like to express their sincere gratitude towards Dr. Alan Tan Wee Chiat for providing background knowledge on the fundamentals of this research.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Chollet, F. *Deep Learning with Python*; Manning Publications Co.: Shelter Island, NY, USA, 2017. Available online: https://www.manning.com/books/deep-learning-with-python (accessed on 6 May 2019).
2. Rabiner, L.R.; Schafer, R.W. *Theory and Applications of Digital Speech Processing*; Pearson: Upper Saddle River, NJ, USA, 2011; p. 64.
3. Graves, A.; Jaitly, N. Towards End-to-End Speech Recognition with Recurrent Neural Networks. In Proceedings of the International Conference on Machine Learning, Beijing, China, 21–26 June 2014; pp. 1764–1772.
4. Miao, Y.; Gowayyed, M.; Metze, F. EESEN: End-to-End Speech Recognition Using Deep RNN Models and WFST-Based Decoding. In Proceedings of the IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU), Scottsdale, AZ, USA, 13–17 December 2015; pp. 167–174. [CrossRef]
5. Rao, K.; Sak, H.; Prabhavalkar, R. Exploring Architectures, Data and Units for Streaming End-to-End Speech Recognition with Rnn-Transducer. In Proceedings of the 2017 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU), Okinawa, Japan, 16–20 December 2017; pp. 193–199.
6. Zhang, C.; Woodland, P. High Order Recurrent Neural Networks for Acoustic Modelling. *arXiv* **2018**, arXiv:1802.08314V1.
7. Yao, K.; Peng, B.; Zhang, Y.; Yu, D.; Zweig, G.; Shi, Y. Spoken Language Understanding Using Long-Short-Term Memory Neural Networks. In Proceedings of the IEEE Spoken Language Technology Workshop (SLT), South Lake Tahoe, NV, USA, 7–10 December 2014; pp. 189–194.
8. Mesnil, G.; Dauphin, Y.; Yao, K.; Bengio, Y.; Deng, L.; Dilek, H.-T.; He, X.; Heck, L.; Tur, G.; Yu, D.; et al. Using Recurrent Neural Networks for SLOT filling in Spoken Language Understanding. *IEEE/ACM Trans. Audio Speech Lang. Process.* **2015**, *23*, 530–539. [CrossRef]
9. Liu, X.; Sarikaya, R.; Sarikaya, R.; Zhao, L.; Pan, Y. *Personalized Natural Language Understanding*; Interspeech: Dresden, Germany, 2016; pp. 1146–1150.
10. Dyer, C.; Ballesteros, M.; Ling, W.; Matthewset, A.; Smith, N.A. Transition-Based Dependency Parsing with Stack Long Short-Term Memory. *arXiv* **2015**, arXiv:150508075v1.
11. Barone, A.V.; Heicl, J.; Sennrich, R.; Haddow, B.; Birch, A. Deep Architectures for Neural Machine Translation. In Proceedings of the Workshop on Machine Translations, Copenhagen, Denmark, 7–11 September 2017.
12. Han, S.; Kang, J.; Mao, H.; Hu, Y.; Li, X.; Li, Y.; Xie, D.; Luo, H.; Yao, S.; Wang, Y.; et al. Ese: Efficient Speech Recognition ENGINE With sparse lstm on Fpga. In Proceedings of the 2017 ACM/SIGDA. International Symposium on Field-Programmable Gate Arrays, Monterey, CA, USA, 22–24 February 2017; ACM: New York, NY, USA, 2017; pp. 75–84.

13. Wang, S.; Li, Z.; Ding, C.; Yuan, B.; Qiu, Q.; Wang, Y.; Liang, Y. C-lstm: Enabling Efficient lstm Using Structured Compression Techniques on Fpgas. In Proceedings of the 2018 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays, Monterey, CA, USA, 25–27 February 2018; ACM: New York, NY, USA, 2018; pp. 11–20.
14. Li, Z.; Ding, C.; Wang, S.; Wen, W.; Zhuo, Y.; Liu, C.; Qiu, Q.; Xu, W.; Lin, X.; Qian, X.; et al. E-RNN: Design Optimization for Efficient Recurrent Neural Networks in FPGAs. *arXiv* **2018**, arXiv:1812.07106.
15. Wani, M.A.; Bhat, F.A.; Afzal, S.; Khan, A.I. Basics of Supervised Deep Learning. In *Advances in Deep Learning*; Springer: Singapore, 2020; pp. 13–29.
16. Ševčík, B.; Brančík, L.; Kubíček, M. Analysis of Pre-Emphasis Techniques for Channels with Higher-Order Transfer Function. *Int. J. Math. Models Methods Appl. Sci.* **2011**, *5*, 433–444.
17. Podder, P.; Khan, T.Z.; Khan, M.H.; Rahman, M.M. Comparative performance analysis of hamming, hanning and blackman window. *Int. J. Comput. Appl* **2014**, *96*, 1–7. [CrossRef]
18. Oppenheim, A.V.; Schafer, R.W. From frequency to quefrency: A history of the cepstrum. *IEEE Signal Process. Magazine* **2004**, *21*, 95–106. [CrossRef]
19. Kumar, K.; Kim, C.; Stern, R.M. Delta-spectral cepstral coefficients for robust speech recognition. In Proceedings of the 2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Brighton, UK, 12–17 May 2011; pp. 4784–4787.
20. Garofolo, J.S. *TIMIT Acoustic Phonetic Continuous Speech Corpus*; Linguistic Data Consortium: Philadelphia, PA, USA, 1993.
21. Chang, A.X.M.; Martini, B.; Culurciello, E. Recurrent neural networks hardware implementation on FPGA. *arXiv* **2015**, arXiv:1511.05552.