Sensitivity Analysis of Gigabit Concept Mining System

Andrew Levine, Ron Loui, John W. Lockwood, Young H. Cho
Reconfigurable Network Group
Washington University in St. Louis
1 Brookings Drive, St. Louis 63130
{aal1, lockwood, young}@arl.wustl.edu, loui@cse.wustl.edu
http://www.arl.wustl.edu/arl/projects/fpx/reconfig.htm

Abstract—As described in our prior papers, we have implemented a system that performs real-time analysis and classification of network traffic using reconfigurable hardware. In this paper, we consider how to optimize the performance and make best use of the hardware resources by simulating the effect of parameter variation. We have devised a systematic method to determine the best parameters for the hardware such that we do not sacrifice the quality of the result. We applied the method to determine how our existing system could best identify the topics of Internet newsgroup postings as the content streams over a Gigabit Ethernet link. [1] [2]

TABLE OF CONTENTS

1. INTRODUCTION

Every day, trillions of packets are transferred across networks throughout the world. To provide security for the users of these networks, network administrators analyze as much content as possible. To be useful, this network traffic must be processed in real-time to avoid accumulating a backlog. Previously, administrators were overwhelmed by massive volumes of data.

To enable processing of large volumes of data, we implemented a hardware-accelerated system called the Automated Front End (AFE) system [1,2]. This AFE system identifies the topic and/or language of network traffic flows as they pass over backbone links. The system outputs scores that indicate proximity to known topics or dialects of a language.

The AFE system is a hardware implementation of a Machine Learning technique. Data is processed at high speed by using Field Programmable Gate Array (FPGA) devices. Use of the FPGA devices enable the system to process and annotate data at Gigabit/second rates. Parameters of the hardware were tuned to provide the best quality of results. Using parameters that find too much data overwhelms an analyst. Using parameters that find too little data is also unsatisfactory because important data may be overlooked.

Several considerations were factored into the design of the AFE system. The AFE hardware was implemented using a stack of 5 FPX platforms. Verification of each FPX module enabled the incremental verification of the entire system. Software tools were written to verify and test each module. These tools used the same data formats used by hardware. By standardizing the interfaces for the components of the AFE system, multiple developers were able to contribute components to the system. Our tools not only enabled us to verify correct operation of the actual AFE hardware that was implemented, but also enabled us to experiment with different variations of the AFE system that used other parameters.

2. BACKGROUND

Field Programmable Port Extender

The AFE system was built using the Field Programmable Port Extender (FPX) platform, an open, reconfigurable hardware platform developed by members of the Reconfigurable Networking Group at Washington University in St. Louis [5,6,7]. Hardware resources on the FPX platform were optimized for processing of data packets and flows passing over Gigabit/second links. The FPX includes two FPGA devices, two banks of Static Random Access Memory (SRAM), and two banks of Synchronous Dynamic Random Access Memory (SDRAM). In total, the FPX hardware can access over a Gigabyte (GB) of memory using the four parallel memory banks. The first FPGA on the FPX, the Network Interface Device (NID), is a Xilinx XCV600E that routes data through the FPX platform, dynamically reconfigures the other FPGA device, and processes commands sent over the network. The second

[1] This research was sponsored by the Air Force Research Laboratory, Air Force Materiel Command, USAF, under Contract Number MDA972-03-9-0001. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of AFRL or the U.S. Government.

IEEEAC paper #1165, Version 17, Updated: 2006:12:20

FPGA, the Reprogrammable Application Device, is a large Xilinx XCV2000E FPGA device that performs reconfigurable data processing as per the configuration of the bitfile loaded by the NID into the RAD.

Circuits for the RAD can be implemented using any standard Hardware Description Language (HDL). For our work, we compiled designs written with the Very high speed integrated circuit HDL (VHDL), synthesized circuits using Synplicity and Xilinx Electronic Design Automation (EDA) tools, and then downloaded bitstreams into the RAD on the FPX.

The complete AFE system was built using a stack of five FPX modules. Each FPX module performed a different function. Together, all of the modules implemented a deep pipeline for packet and flow processing.

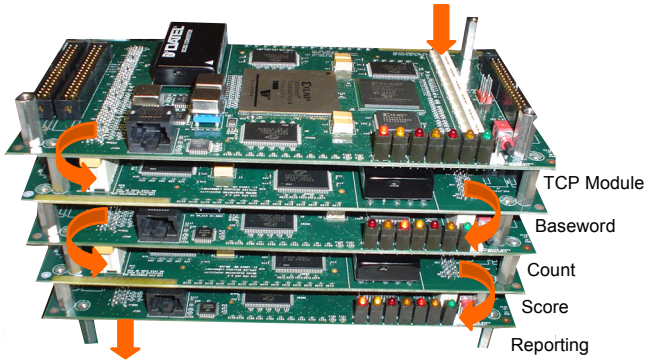

Figure 1 – A stack of five Field-programmable Port Extender (FPX) platforms built with a total of ten FPGAs (5 RADs and 5 NIDs) implement Automated Front End (AFE) processing Hardware.

Circuits that run on the FPX exploits parallel hardware to achieve Gigabit/second data processing rates. The FPX platform has been used to build dozens of high-speed networking applications, but is especially well suited for the task of deep packet inspection. By using hardware, rather than software, the FPX platform can process data at rates that far surpass conventional computing systems. A single FPX hardware module can replace a rack full of PCs. Unlike an Application Specific Integrated Circuit (ASIC), the FPGA bitfiles on the FPX platforms can be dynamically reconfigured to perform different processing tasks.

Configuration of the System

The configuration of the hardware for the AFE system is shown in Figure 1. The system is implemented using a stack of five FPX cards which are individually configured to (1) analyze TCP/IP flows, (2) identify basewords that appear in text, (3) count the frequency of the basewords, (4) score the data to known concepts, and (5) report the score to an external software system.

Data is received by the system cards from a Line Card (LC) that stacks on top of the FPX modules. Two types of LCs were built: one for OC48 and the other for Gigabit Ethernet.

A chassis holds the stack of FPX cards, the configuration is shown in Figure 2. In this AFE system, four FPX cards are stacked on the left side of a chassis below the LC to reconstruct the traffic flows and process the streams of data. The stacked FPX cards are connected together via high-speed Utopia interfaces and data flows between cards using virtual circuits. Physically, each FPX card is connected to other FPX modules above and below that card. A backplane interconnects a stack of FPX modules on the left and right side of the chassis. A fifth FPX card (called the NID PT) is placed on the right side of the chassis. This module connects to a Gigabit Ethernet LC to report a summary of each flow to an external software system. The final output of the hardware is transmitted using standard UDP/IP packets which are in turn received by a server that "catches" the results.

The processing stages of the AFE system are implemented with a deep pipeline broken down to basic functions. First, the TCP Circuit extracts payloads from network traffic. Second, the Base Word Circuit analyzes payloads by identifying acceptable byte sequences and passing them through a dimensionality reduction system called the Word Mapping Table (WMT). Third, values are passed to the Count Circuit which builds a representation of the document into an array. Finally, the Score Circuit completes a dot product with preloaded concepts represented by arrays of values. The output of the score circuit is passed to the NID PT which in turn writes out summaries of the flows to the hardware Gigabit LC [1].

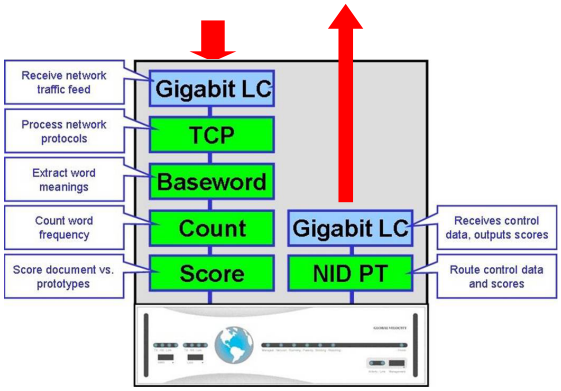

Figure 2 Processing Stack of FPX Cards

3. HARDWARE SEMANTIC PROCESSING

Although reconfigurable hardware can process data at high speeds, certain considerations must be taken into account when implementing modules with FPGAs. For example, floating point computation is possible but it is costly in terms of the resulting size of hardware circuits. A floating point multiplier, for example, is four times larger than an

integer multiplier. Implementing a few floating point units is not a problem, but does not make the best use of the hardware's resources. For semantic processing, a large number of features is needed to represent a document. Therefore, a large number of multipliers would require a large amount of hardware. Fixed-size multipliers are much better suited for the task than a floating point unit.

Extracting Features - The Base Word Circuit

Feature extraction is performed in the Base Word Circuit. The Base Word Circuit considers a word to be a series of bytes that are identified as acceptable in a given sequence. Acceptance of byte sequences is different than tokenization. Tokenization of words is performed by searching for tokens and segmenting sequences of bytes at those known tokens whereas the identification of words from acceptable byte sequences allows sequences to be built from any pattern that adheres to acceptable values.

The AFE system can process words in different languages by supporting characters that are either a one or two bytes long. For the AFE system, any sequences of 3 to 16 characters can constitute an acceptable word. Sequences that extend past 16 characters are truncated to a length of 16. For example, a long English word (using single-byte encoding) is truncated to the shorter 16-byte string as shown below:

antidisestablishmentarianism → antidisestablish

Minimum Word Length

The AFE system can be set to process only base words that have a minimum length. By default, we process data with word lengths that have a minimum of 3 characters. For our analysis, however, we varied the minimum word lengths from 2 to 8 characters.

After a word is identified, the AFE system hashes that word into a 20-bit field that has a range of 0 to 1,048,575. The Word Mapping Table (WMT) uses this 20-bit value to index a memory location in an SRAM attached to the FPX card. Each element in the memory holds a smaller value which falls in the range of 0 to 3,999--the default dimensions that we use to represent a document. This content of the Word Mapping Table is programmable by software. An example of how input words are hashed into a 20-bit value then mapped into a baseword in the range of 0 to 3,999 is shown in Figure 3. The output from the Baseword module produces a list of features that appear in the documents.

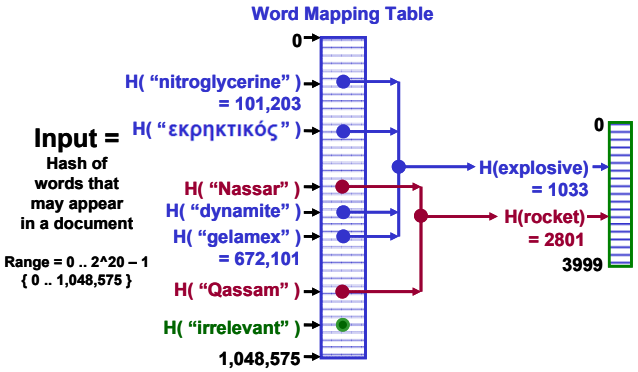

Figure 3 Word Mapping Table in Base Word Circuit

Dimensionality – Number of Features

In this work, we considered the effect of using fewer dimensions to represent features. By default, the AFE system uses 4,000 bins to represent the documents and concept vectors. We varied the parameters to determine the effect of using dimensions: 3,000, 2,000, 1,000, 500, and 250 dimensions.

Building Representation - The Count Circuit

The feature counting function is performed by the Count Circuit. The Count Circuit receives a list of extracted features and increments the appropriate counters for the document. The circuit accumulates statistics of words that appear in the document by tracking the counts for each flow. The hardware uses integers to track the counts. As one of the design considerations of the AFE system, 2 KBytes of SRAM are reserved for each vector because this is the amount of space needed to represent 4,000 feature vectors with 4 bit counters. When a document is finished counting, the vector is passed down the pipeline to the Score module.

Count Bit Resolution

The Count module in the AFE system uses an array of saturating, finite-sized counters to track the frequency of occurrence for each baseword. In this analysis, we consider the effect of using fewer bits for the count of each feature. By default, the resolution of the bins in the document's count arrays is nominally 4 bits allows for a range of 0 to 15 in each bin. Through analysis in this paper, we also consider counters that use: 3 bit, 2 bit, and 1 bit resolution.

Scoring Documents - The Score Circuit

Scores are computed as a dot product computation between a document's representation generated by the count circuit and known concepts. The concepts are represented with the same number of feature dimensions and either 4-bit or 8-bit integer counters. With 4,000 dimensions, 2KB or 4KB are needed to represent a concept depending on whether the size of each feature is a half byte or a full byte. The Score Circuit computes a dot product of the document against all the loaded concepts as seen in Figure 4. When the

computation is complete, the results are passed out of the system.

Score Bit Resolution

In this work, we considered variations of the system with 8 bits or 4 bits of resolution. Hardware circuits were built that supported both values of resolution. A tradeoff exists with hardware because more concepts can be represented using a smaller resolution.

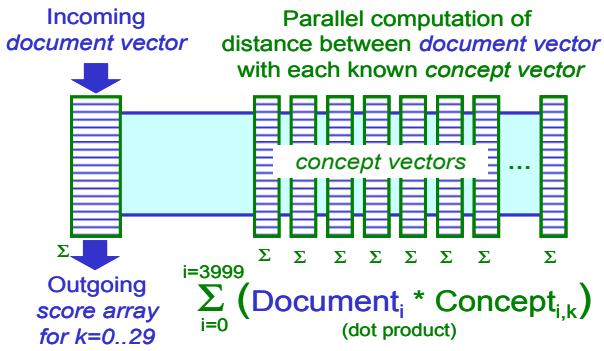

Figure 4 Scoring Documents in the Score Circuit

4. Parameter Manipulation Experiment

Parameter Variation

The parameters of the AFE system were varied as follows: there were a total of 7 values varied in minimum word length size, 4 values varied in count bin resolution, 2 values varied in score table bin resolution, and 6 values varied for the number of dimensions for a total of (7 x 4 x 2 x 6) 336 combinations. In order to obtain a statistically accurate set of results from a specific combination of the parameters, 30 individual runs were constructed. With 30 runs per configuration, the total number of experimental runs is 10,080. The parameter manipulation experiment considers the quality of the result using both the Mutual Information (MI) and the Heuristic [1, 8, 10] algorithms that develop the WMTs and STs. Considering both algorithms, a total of 20,160 variations of the system were simulated.

Corpus Profile

The corpus used in the experiment is from a number of Usenet news groups retrieved from an archive on Google [11]. The profile of the corpus is described in Figure 5. The last three columns show the number of randomly selected documents used for training and testing. Headers were stripped so that the system could not simply train and classify documents using labels.

Google Group	Label	Training	Testing	Total
sci.archeology.moderated	archaeology	70	70	140
alt.sports.baseball.stl_cardinals	baseball	32	34	66
rec.equestrian	equestrian	41	42	83
misc.consumers.frugal_living	frugal	16	18	34
soc.libraries.talk	libraries	16	18	34
sci.logic	logic	31	31	62
rec.martial_arts.moderated	martial_arts	28	30	58
comp.ai.neural_nets	neural_nets	23	26	49
comp.programming.threads	programming	47	48	95
humanities.music.wagner	wagner	29	32	61
misc.writing.moderated	writing	37	38	75
talk.origins	chaff	368	10402	10770
	Total:	738	10789	11527

Figure 5 Corpora Selection

All files utilized in the experiments had at least 200 words of text and newsgroup headers stripped. The total set of documents was divided in half for training and testing in all groups except the "chaff" group. The "chaff" group was used to provide interference by containing valid data that did not relate to other topics. A subset of the chaff articles was used in training so that chaff documents would fall into the same category during testing. We assumed that when large amounts of data were processed, the ratio of interesting to uninteresting items would be small. By training on chaff, we provided an attractor class for undesired items and reduced the rate of false positives.

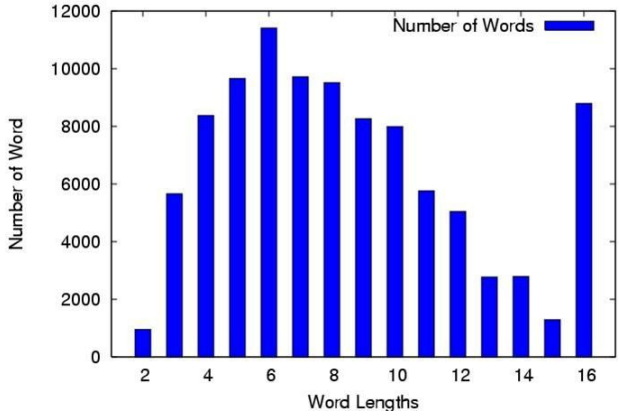

Figure 6 Minimum Word Lengths

An experiment was performed to test the precision of the AFE system while manipulating the parameters of the simulated hardware. By increasing the minimum word length, the number of words derived from the training set of documents decreases. When the minimum word length is too short, however, many words are rejected by the algorithms as insignificant. This distribution of word lengths is shown in Figure 6.

Performance Metrics

Performance of the classification system is defined by the use of the metrics precision and recall. For a given class, precision is defined as the number of documents correctly retrieved in that class divided by the total number of documents retrieved, whether correctly assigned to the class or not.

C = set documents for a single class

D = number documents in C retrieved

F = number documents not in C retrieved

D + F = total number documents in classification

Precision of C = D/(D + F)

For a given class, Recall is defined as the ratio of the number of documents correctly retrieved in that class, divided by the total number of documents that could have been correctly retrieved in the class.

C = class of documents

D = number documents C retrieved

E = number documents in C not retrieved

|C| = D + E (total number of documents in C)

Recall of C = D/(D + E)

Thresholds for Less Confusion

Thresholds were used in the classification of documents for individual runs. In the training phase of a run, the least correct classification of a document was used as a minimum for a classification in the testing phase. Thresholds insure that there is a minimum score required for classification. Forcing a classification of all documents will lead to misclassification. The end user of the AFE system should not have to sift through huge volumes of unwanted data. The use of a threshold for classification insures that a document is only classified if there is high confidence that it matches a concept.

Results of Analysis

In the analysis of the experiment, the optimal thresholds for each run were determined individually. A confusion matrix was calculated for both the training data and the test data. For example, Figure 7 shows a confusion matrix for a run of the MI algorithm with the parameter setting of: minimum word length = 5, count bit resolution = 2, score table resolution = 8, and dimensionality = 3000. Figure 8 shows the confusion matrix for the testing side of the run. No documents used for training were used in the data processed testing. Note that the system maintains good classification as evidenced by the large numbers of correctly classified documents on the diagonal.

	archaeology	baseball	equestrian	frugal	libraries	logic	martial_arts	neural_nets	programming	wagner	writing	chaff	reject	recall
	70	0	0	0	0	0	0	0	0	0	0	0	0	100
	0	32	0	0	0	0	0	0	0	0	0	0	0	100
	0	0	41	0	0	0	0	0	0	0	0	0	0	100
	0	0	0	16	0	0	0	0	0	0	0	0	0	100
	0	0	0	0	16	0	0	0	0	0	0	0	0	100
	0	0	0	0	0	31	0	0	0	0	0	0	0	100
	0	0	0	0	0	0	28	0	0	0	0	0	0	100
	0	0	0	0	0	0	0	23	0	0	0	0	0	100
	0	0	0	0	0	0	0	0	47	0	0	0	0	100
	0	0	0	0	0	0	0	0	0	29	0	0	0	100
	0	0	0	0	0	0	0	0	0	0	37	0	0	100
	0	0	0	0	0	0	0	0	0	0	0	368	0	100
precision	100	100	100	100	100	100	100	100	100	100	100	100		

Figure 7 Training Confusion Matrix for Mutual Information Algorithm, minimum word length 5, count bit resolution 2, score table resolution 8, and 3000 dimensions

	archaeology	baseball	equestrian	frugal	libraries	logic	martial_arts	neural_nets	programming	wagner	writing	chaff	reject	recall
	17	0	0	0	0	0	0	0	0	0	0	51	25.00	
	0	18	0	0	0	0	0	0	0	0	0	14	56.25	
	0	0	11	0	0	0	0	0	0	0	0	29	27.50	
	0	0	0	0	0	0	0	0	0	0	2	14	0.00	
	0	0	0	0	0	0	0	0	0	0	5	11	0.00	
	0	0	0	0	0	14	0	0	0	0	2	13	48.28	
	0	0	0	0	0	0	15	0	0	0	2	11	53.57	
	0	0	0	0	0	0	0	7	0	0	3	14	29.17	
	0	0	0	0	0	0	0	0	37	0	0	9	80.43	
	0	0	0	0	0	0	0	0	0	9	0	4	17	30.00
	0	0	0	0	0	0	0	0	0	0	13	2	21	36.11
	1	0	0	0	0	1	2	0	0	0	0	9389	1007	90.28
prediction	94	100	100	0	0	93	88	100	100	100	100	99		

Figure 8 Testing Confusion Matrix for Mutual Information Algorithm, minimum word length 5, count bit resolution 2, score table resolution 8, and 3000 dimensions

Averaging Runs

The last columns of Figures 7 and 8 show the recall for documents in each class. The last row shows the precision. In order to determine how well the algorithm performed, the averages of the precision and recall were computed over all 30 runs. The average was computed for each class, irrespective of the size of the class. The average recall for the test set in Figure 8 is 39.72%. The average precision of the test set in Figure 8 is 81.32%. For the Heuristic algorithm, this run had an average recall value of 34.74% and an average precision of 79.37%. Once the average of each individual run is calculated, the average over the 30 runs is calculated with a 95% confidence interval. A confidence interval is calculated by:

95% confidence interval = $1.96 \bullet \sigma_m$

$\sigma_m = \sigma/\sqrt{N}$ is the standard error of the mean.

An example of the plot for a dimensionality of 3000 is shown in Figures 9 and 10.

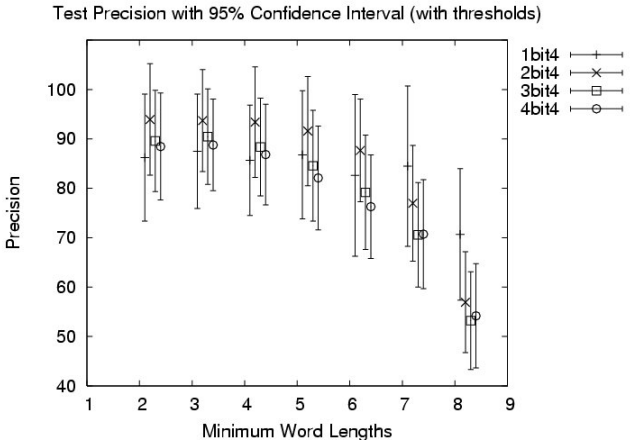

Figure 9 Precision Results for Mutual Information Algorithm using 3000 Dimensions and 4 bit Score Table Resolution

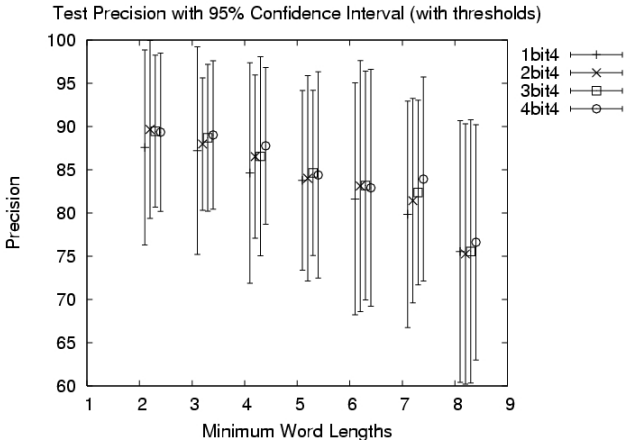

Figure 10 Precision Results for Heuristic Algorithm using 3000 Dimensions and 4 bit Score Table Resolution

5. RESULT ANALYSIS

When viewing all the graphs of the run, it is possible to see where the dimensionality changes start to have a substantial effect on the performance of the algorithms. When combining the results for all dimensions in Figures 11 and 12, the inverse relationship between precision and recall can be seen most clearly in plots of the Heuristic algorithm runs. However, viewing the MI run results in Figures 13 and 14 shows little separation of results.

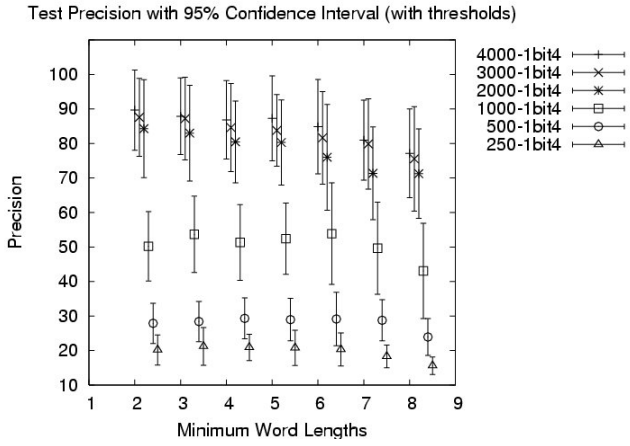

Figure 11 Heuristic Algorithm Precision Results for 1 bit Count Resolution

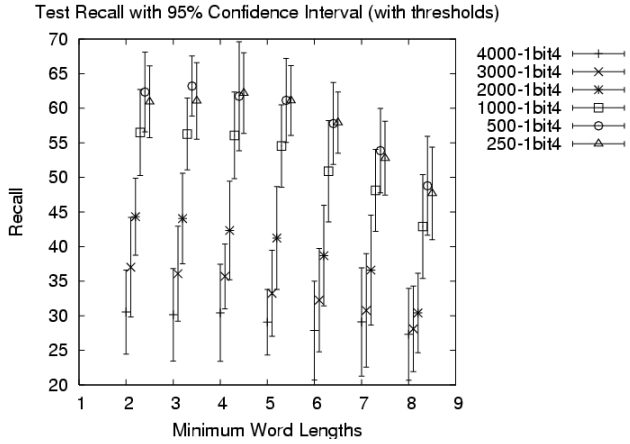

Figure 12 Heuristic Algorithm Recall Results for 1 bit Count Resolution

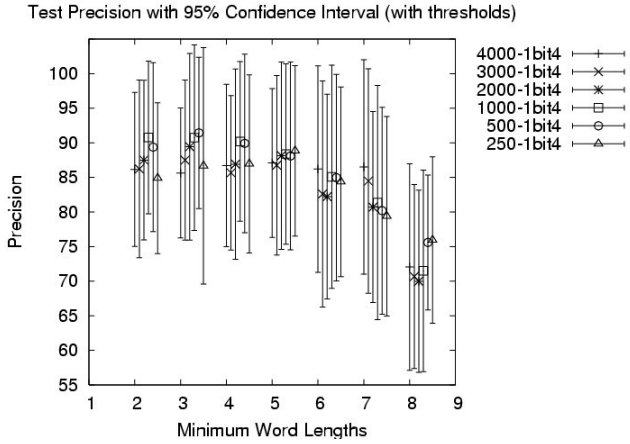

Figure 13 MI Algorithm Precision Results for 1 bit Count Resolution

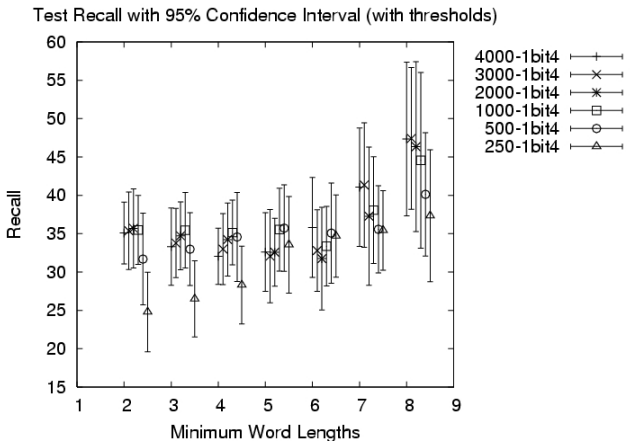

Figure 14 MI Algorithm Recall Results for 1 bit Count Resolution

Minimum Word Length

Consistently, a trend can be observed showing that as the minimum word length increases, the precision decreases. Conversely, recall goes up as the minimum word length increases. However, the values in the range of 2, 3, 4, and 5 seem exceptionally close to each other in all configurations. Both the MI and Heuristic algorithms parallel each other when considering only a change in the minimum word length. There is an incentive to limit processing of short words so as to save the system from performing extra work.

The AFE system looks at 4 bytes per clock cycle. In terms of processing requirements, the worst case occurs with a minimum word length of 2. A sequence of text with short words would generate 4 words every 3 clock cycles. Therefore, it is possible to produce more than one word per clock cycle. This could be handled by a circuit, but the system would be over designed with no need for it. With a minimum word length of 3, words could be produced as often as every clock cycle. Although it is possible for the count circuit to handle this load, as the system currently does, with the goal of making the system faster, the creation of base words is a detriment to performance. A small minimum word length forces multiple clock cycles to be used in order to produce a word.

When choosing a minimum word length of 3 bytes, the circuit need only be designed to handle a new word every clock cycle. If a minimum word length of 4 or 5 is chosen, fewer words need to be counted. It is also the case that most short words are stop words that should be ignored for semantic analysis. Therefore, selecting a minimum word length of 4 has an added advantage.

There are other motivations for using larger minimum word lengths. It is not desirable to find acceptable strings in binary files. The AFE Base Word Circuit does not confirm that words fall into a dictionary. Words like “ZKT” are treated the same as words like “YES” if they happen to hash to the same base word table entry. It is also possible that these nonsense words can produce the same base word as a word like ``RIVER'' with a hash collision. Reducing noise is one of the benefits of increasing minimum word length.

Number of Features

The degraded separation that is seen when changing dimensions in the Heuristic algorithm remains when count bit and score bit resolutions are increased. The confusion matrices for the Heuristic algorithm show that the precision is low because chaff is misclassified. This occurs even more often when the parameters of the system are set so that there are 500 or 250 bins for features. The Recall for those runs is high. However, if the goal of the system is to present data with high precision, limiting the number of features for the Heuristic algorithm becomes an issue. The effect of limiting features affects the precision of the Heuristic algorithm when the number of features drops from 2,000 to 1,000. The runs with 4,000, 3,000, and 2,000 feature results are close for all other parameter manipulations. But, 2,000 features appear to be the lower end of the number of features necessary to maintain high precision. Using less than 2,000 features significantly degrades the Heuristic algorithm.

Results for the MI algorithm show that the change in the number of features is not as much of an issue for the performance of the algorithm. The results show that the MI is not affected much when the number of features was reduced to 2,000. The reason is that the MI algorithm rarely uses more than 1,500 features with the training data in these experiments.

Count Bit Resolution

Results from running the Heuristic algorithm show only minimal differences in precision occur when the count bit resolution changes. This effect can be observed in Figures 15 and 16. The MI algorithm does show minimal degradation as a result of changes to the count bit resolution. Figures 17 and 18 show how changes have a greater affect on the performance of the system. Consistently, for the MI algorithm, 2 bit count resolution works slightly better than the other values.

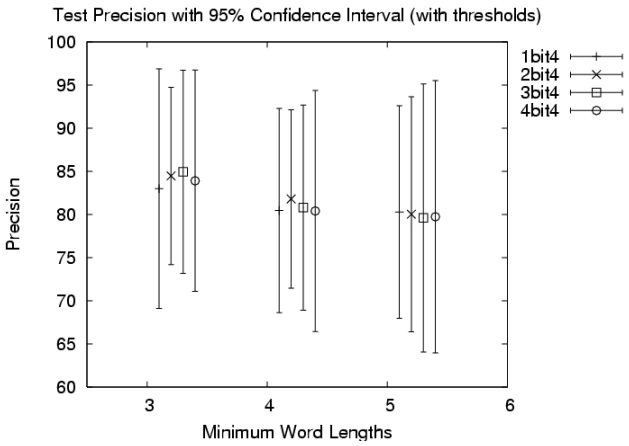

Figure 15 Heuristic Algorithm Precision Results for 2,000 Dimensions and 4 bit Score Table Resolution

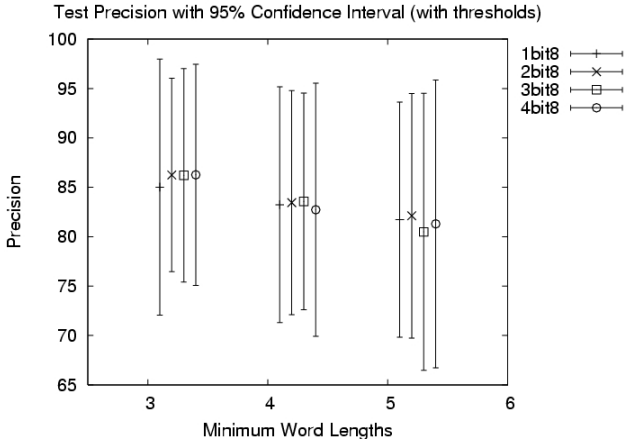

Figure 16 Heuristic Algorithm Precision Results for 2,000 Dimensions and 8 bit Score Table Resolution

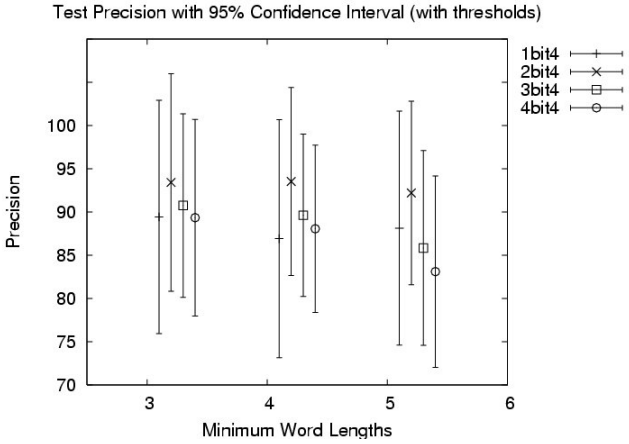

Figure 17 MI Algorithm Precision Results for 2,000 Dimensions and 4 bit Score Table Resolution

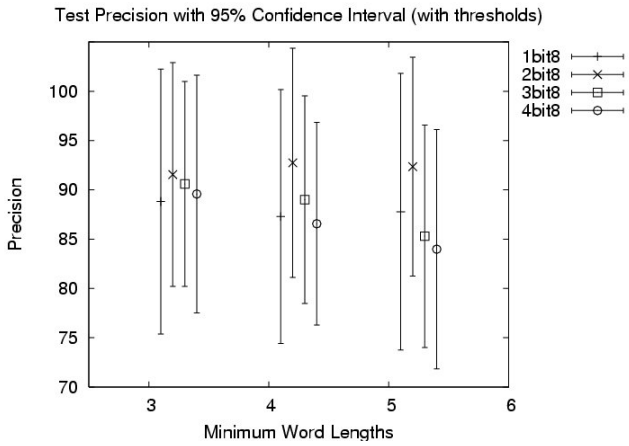

Figure 18 MI Algorithm Precision Results for 2,000 Dimensions and 8 bit Score Table Resolution

6. SIGNIFICANCE EVALUATION

We ran an experiment to find a set of parameters that required fewer hardware resources while not losing precision. In an experiment with 12 newsgroups, the choice of minimum word length of 4, 2,000 features, 2 bit count bit resolution, and 4 bit score bit resolution were shown to not lessen the precision of the system. When using the MI algorithm, the average precision of the original configuration is 89.3% with a standard deviation of 29.2. The proposed setting has a precision of 93.5% with a standard deviation of 30.8. The proposed setting has a higher average precision with a slightly higher standard deviation. It would appear that the changes did not degrade performance, but rather improved it. However, looking at the results from the Heuristic algorithm, the original setting yielded an average precision of 91.3% with a standard deviation of 28.2. This is in contrast to the proposed setting that had an average precision of 81.8% with a standard deviation of 27.8. For the Heuristic algorithm, the change in settings appears to be detrimental to precision.

To test the significance of the results, a Paired T test is used. The results from the two settings provide a distribution of results and the test indicates the amount of difference between the two results. The evaluation takes the form:

$$\text{Paired T Test Statistic} = D/(S_d/\sqrt{n})$$

where D is the average of the differences at each data point. S_d is the standard deviation of the differences and n is the number of data points (30). The results from the Heuristic algorithm are compared for the original and proposed settings. The D value is 9.50 and the S_d value is 6.11 which results in the value of 8.51. The P value for $t_{0.0005}$ is 3.646 which shows that there is a statistical difference when changing the settings of the AFE. Applying the same T test to the MI results shows D is -4.28 and S_d is 5.82. This results in a value of -4.03 which indicates a significant

difference in the results of the system. For the MI algorithm, the performance increases and for the Heuristic algorithm it decreases. Performing the T test between the results of the two algorithms shows that there is no difference between the two algorithms with the original settings of the system. The T test at the original settings has a value of 1.90 which does not beat the criteria of 3.646. In comparing the performance of both algorithms with the proposed settings, the T test has a value of 10.84. It is clear that the MI algorithm achieves greater precision at the new settings.

7. CONCLUSION

In this paper, a framework was developed to determine the optimal parameters for representation of flows for classification. The numeric representation of a flow vector affects the area in the FPGA and computational processing time. By finding optimal parameters for the AFE system, the precision in classification is maintained while minimizing the cost to implement the system.

The experiment performed within this paper shows that for 12 newsgroups, the size of the vectors needed to represent flows can be reduced by 75% without loss of precision. The technique used for the parameter experiment can be utilized to obtain settings for other types of data. Since the AFE system makes use of a pipeline for processing, other types of processing circuits could be inserted into the pipeline. Metadata, for example, could make use of the area saved by the reduced space needed to classify text.

REFERENCES

[1] J. W. Lockwood, S. G. Eick, J. Mauger, J. Byrnes, R. P. Loui, A. Levine, D. J. Weishar, and A. Ratner, "Hardware Accelerated Algorithms for Semantic Processing of Document Streams", 2006 IEEE Aerospace Conference, March 4-11, 2006.

[2] J. W. Lockwood, S. G. Eick, D. J. Weishar, R. P. Loui, J. Moscola, C. Kastner, A. Levine, and M. Attig, "Transformation Algorithms for Data Streams, 2005 IEEE Aerospace Conference", March 5-12, 2005.

[3] J. Byrnes and R. Rohwer, "An Architecture for Streaming Coclustering in High Speed Hardware", IEEE Aerospace Conference, March 4-11, 2006.

[4] C. Kastner, G. A. Covington, A. Levine, and J. Lockwood, "HAIL: A Hardware-Accelerated Algorithm for Language Identification", 15th Annual Conference on Field Programmable Logic and Applications (FPL), August 24-26, 2005.

[5] J. W. Lockwood, "Evolvable Internet Hardware Platforms", NASA/DoD Workshop on Evolvable Hardware (EHW'01), July 2001.

[6] John W. Lockwood, Jon S. Turner, David E. Taylor, "Field Programmable Port Extender (FPX) for Distributed Routing and Queuing", ACM International Symposium on Field Programmable Gate Arrays (FPGA'2000), February 2000.

[7] John W. Lockwood, Naji Naufel, Jon S. Turner, David E. Taylor, "Reprogrammable Network Packet Processing on the Field Programmable Port Extender (FPX)", ACM International Symposium on Field Programmable Gate Arrays (FPGA'2001), February 2001.

[8] J. B. MacQueen, "Some Methods for Classification and Analysis of Multivariate Observations", 5th Symposium on Mathematics, Statistics and Probability, 1967.

[9] G. A. Covington, C. Comstock, A. Levine, J. Lockwood, and Y. H. Cho, "High Speed Document Clustering in Reconfigurable Hardware" 16th Annual Conference on Field Programmable Logic and Applications, August 28-30, 2006.

[10] J. Byrnes and R. Rohwer, "Text Modeling for Real-Time Document Categorization", IEEE Aerospace Conference, March, 2005.

[11] Google Inc. Google groups. http://groups.google.com/, 2005

BIOGRAPHY

Andrew Levine is a graduate student in the Reconfigurable Network Group at Washington University in St. Louis. His primary interest is in Machine Learning techniques for classification and clustering. Currently, he is focusing on streaming data algorithms for classification and concept discovery. He is intending to pursue a Ph.D. in Artificial Intelligence.

John W. Lockwood designs and implements networking systems in reconfigurable hardware. He leads the Reconfigurable Network Group (RNG) at Washington University. The RNG research group developed the Field programmable Port Extender (FPX) to enable rapid prototype of extensible network modules in Field Programmable Gate Array (FPGA) technology. He is an Associate professor in the Department of Computer Science and Engineering at Washington University in Saint Louis. He has published over 75 full-length papers in journals and major technical conference proceedings that describe technologies for providing extensible network services in wireless LANs and in high-speed networks. Professor Lockwood has served as the principal investigator on grants from the National Science Foundation, Xilinx, Altera, Nortel Networks, Rockwell Collins, and Boeing. He has worked in industry for AT&T Bell Laboratories, IBM, Science Applications International Corporation (SAIC), and the National Center for Supercomputing Applications (NCSA). He served as a co-founder of Global Velocity, a networking startup company focused on high-speed data security. Dr. Lockwood earned his MS, BS, and Ph.D degrees from the Department of Electrical and Computer Engineering at the University of Illinois. He is a member of IEEE, ACM, Tau Beta Pi, and Eta Kappa Nu.

Ron Loui is an Associate Professor in Computer Science and Engineering. He is the author of over seventy articles in leading technical journals published over the past two decades including AI Journal, Cognitive Science, Computational Intelligence, Journal of Philosophy, Journal of Symbolic Logic, MIT Encyclopedia on Cognitive Science, AI and Law, Theory and Decision, CACM, and ACM Computing Surveys. He was a Stanford Sloan Fellow and received his undergraduate degree at Harvard with high honors in Applied Mathematics: Decision and Control, 1982. He received a joint Computer Science and Philosophy doctoral degree from the University of Rochester after a MS, in 1987.

Young Cho is a Visiting Assistant Professor in the Computer Science and Engineering Department of Washington University in St. Louis. He has earned his BA in Computer Science from UC Berkeley while working under the supervision of Prof. David Patterson and Prof. David Culler, MSE in Computer Engineering at UT Austin under Prof. Brian Evans, and Ph.D in Electrical Engineering at UCLA under Prof. Mangione-Smith. He has also worked under the supervision of Dr. Charles Seitz and Dr. Danny Cohen at a high performance interconnect company, Myricom, Inc. He has designed and implemented a number of high performance research and development projects as well as commercial products during his career. His areas of expertise include network security, computer networks, high performance computer architecture, and reconfigurable computers. He is a member of IEEE and ACM.