Low-Power Portable Devices for Metagenomics Analysis: Fog Computing Makes Bioinformatics Ready for the Internet of Things

Ivan Merelli[a], Lucia Morganti[b], Elena Corni[b], Carmelo Pellegrino[b], Daniele Cesini[b], Luca Roverelli[c], Gabriele Zereik[c], Daniele D'Agostino[c]

[a] *Institute for Biomedical Technologies, National Research Council of Italy, Segrate (Milan), Italy*
[b] *CNAF Section - Italian Institute for Nuclear Physics, Bologna, Italy*
[c] *Institute for Applied Mathematics and Information Technologies "E. Magenes", National Research Council of Italy, Genoa, Italy*

Abstract

Portable sequencing machines, such as the Oxford Nanopore MinION, are making the genome sequencing ubiquitous. This can be particularly interesting for identifying specific bacteria in air-filters or waters and for monitoring the microbioma composition in cultivated soils or in different animal samples, using a simple and portable approach. However, a main problem of these portable sequencing devices is that they stream huge amounts of data, which management can be actually challenging. Low-power System-on-Chip architectures represent a feasible way for designing a solution, based on the Fog computing paradigm, for processing locally the raw data, considering both the base calling step and the genome alignment part, and for sending only meaningful results over Internet. Cloud services can be then used to collect and integrate results in a Internet of Things framework, in order to trigger notifications or alarms and, in perspective, for more sophisticated applications based on statistical or machine learning approaches.

Keywords:
Metagenomics; Environmental genomics; Fog computing; Machine Learning; Cloud computing; Internet of Things;

1. Introduction

The Internet of Things (IoT) paradigm owes its explosive growth to the ability of connecting physical devices (i.e., the *things*) to analytics and machine learning applications, which can help to get insights from device-generated data and enable devices to make smart decisions without human intervention [1]. In order to exploit IoT at full power, high-speed data processing, scalable analytics, and shorter response times are fundamental.

Meeting these requirements is somewhat problematic through the current centralized Cloud-based model powering IoT systems. Therefore, the idea to create a logical and efficient spot in the continuum between the data source and the cloud has been proposed. This paradigm is usually called Fog computing [2, 3], meaning a decentralized architectural pattern that brings computing resources and application services closer to the edge of the IoT network, where the data are being actually generated. Moreover, Fog computing has the key advantage of reducing the amount of data to be transferred to Cloud data centres for processing and analysis, while also improving security, a major concern in the IoT industry [4].

The Fog layer is the perfect junction where there are enough computing, storage, and networking resources to support the local ingestion of data, as soon as they are acquired, and the quick turnaround of results. In most cases, low-power System-on-Chip devices (SoCs) are exploited, because they are designed to meet the best trade off between computing performance and power consumption. Cloud servers, on the other hand, have the horsepower to perform complex analytics and machine learning tasks to integrate time series produced by a number of heterogeneous *things*.

In particular, portable genome sequencing machines as the Oxford Nanopore MinION devices [5] can be used to monitor the bacterial communities of different environments, providing useful information about their interaction with human activities. For example, they can be used to monitor bacteria in air-filters of hospitals, food industries, and pharmaceutical companies in order to give alarms in case some pathogens are identified in the environment [6, 7]. Moreover, they can be used for monitoring the pollutions of waters in sees, lakes and aqueducts [8] and the bacterial communities in cultivated soil [9, 10] and greenhouses, or to test different animal samples from farms [11]. Other more extreme usages of Oxford Nanopore devices have been experimented [12, 13, 14].

So far, the analysis processes are manually conducted, because they are basically spot analyses. With the further development of this technology, a number of these devices will be exploited on the field in a routinely way. Therefore the Fog computing is a natural candidate for designing efficient and effective solutions to manage the results. The key aspect is represented by the possibility of locally processing the huge quantity of data these devices can produce, in order to sent to Cloud applications only the most important results.

From the computational point of view, the interesting feature of Oxford Nanopore devices is that sequencing data starts streaming immediately, rather than being delivered in bulk at the end of a sequencing run. Real-time data streaming allows immediate analysis of the data, enabling sequence mapping as quickly as the results become available, even during the experiment. Rapid identification of organisms in samples is therefore possible, since users recently reported the identification of bacteria and markers of antimicrobial resistance within 5-10 minutes [15].

In this paper, we apply the Fog computing paradigm for designing a system able to support a network of MinION devices working on the field (Figure 1). Our results demonstrate that it is possible to manage the data stream processing, i.e., the possibility to perform base calling and bacteria identification, for real-time analysis with a Fog-based infrastructure making use of SoCs. The present prototype is able to trigger notifications in case of dangerous situations, and we are working on other applications. In fact, when all the data of the MinION network is collected in the Cloud environment, it is possible to address more complex scenarios, as the analysis of bacteria communities with the application of machine learning techniques on the time series.

To the best of our knowledge, no previous works have been published about Fog computing applications in Bioinformatics, while energy efficiency aspects have been considered [16, 17, 18] and the use of Cloud computing is a well established technology [19, 20, 21, 22, 23]. The use of SoC devices for Fog computing applications, instead, is receiving an increasing attention [24, 25, 26] in these last years.

This paper is organised as follows: Section 2 describes the hardware components of the considered system, while Section 3 gives an overview of the workflow for data processing and the IoT infrastructure; Section 4 describes the achieved experimental results, followed by conclusions and future directions.

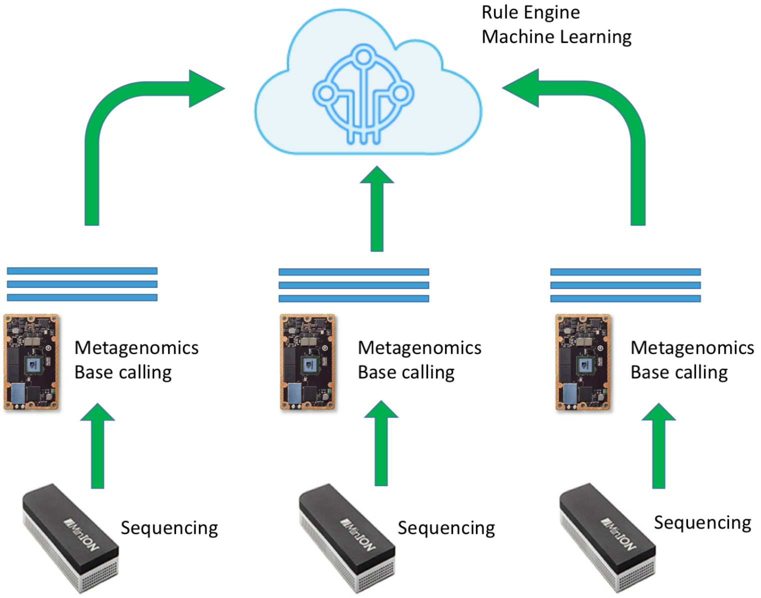


Figure 1: Diagram of the computation. Data are generated from the Oxford Nanopore MinION devices. The Fog layer, based upon SoC devices, is used to perform the base calling and bacteria identification. The results are uploaded to a Cloud infrastructure to be integrated in a machine learning framework, for triggering alarms or give decision feedback to the network.

2. The Biological device

MinION by Oxford Nanopore is a miniaturized sequencing instrument device, with a weight under 100g and powered by its USB port, that can be transported into the field [27]. It basically consists of a set of nanopores, which are nano-scale protein channels in a membrane surface. Each nanopore is powered by a ionic current, which is modulated by biological molecules passing through it, producing a real-time data streaming during the experiment. The current variations can be used to identify the biological molecule passing through the nanopore, that is one of the nucleotides (Adenine, Guanine, Cytosine, Thymine/Uracile) composing the DNA/RNA sequence.

Before sequencing, the sample has to be prepared, which means that DNA/RNA must be extracted and chemically predisposed for sequencing. The actual protocol for library preparation is possible in 5 minutes and using only a minimum of equipment, making the portability of this device, at least from the experimental point of view, the best possible. Moreover, Oxford Nanopore announced the release of an automated sample and library preparation devices, called VolTRAX and shown in Figure 2, relying on a sophisticated micro-fluidics approach, to simplify the workflow even further and enabling its usage without direct human intervention.

The MinION device produces very long reads (reads over 100 Kilo base pairs - Kbp - were reported), although it suffers from high sequencing error rate, up to 15-18%. This can be corrected with bioinformatic approaches, in particular using machine learning techniques. DNA base calls are usually performed using a Cloud-based software provided by Oxford Nanopore, called Metrichor, that employs Hidden Markov Models (HMMs) to infer sequences from these ionic current changes [28].

Although this approach provides good performances, a broadband Internet connection is required, since basically all the raw data (electric signals) produced by the sequencer have to be uploaded to the Cloud. The device has a declared peak throughput of 5-10 Gbp in 48 hours, resulting in about the same amount of GB, even if the real throughput is of usually about 0.5-2.5 Gbp, considering the same time interval [29]. Moreover, results may be given back from Metrichor, in terms of raw DNA/RNA sequences, for further analysis, producing other intensive I/O, which can be very difficult to manage on the field, thus partially impairing the portability of the device.

From the metagenomic perspective, some works have been already done with MinION. To complete real-time environmental genomic experiments, a

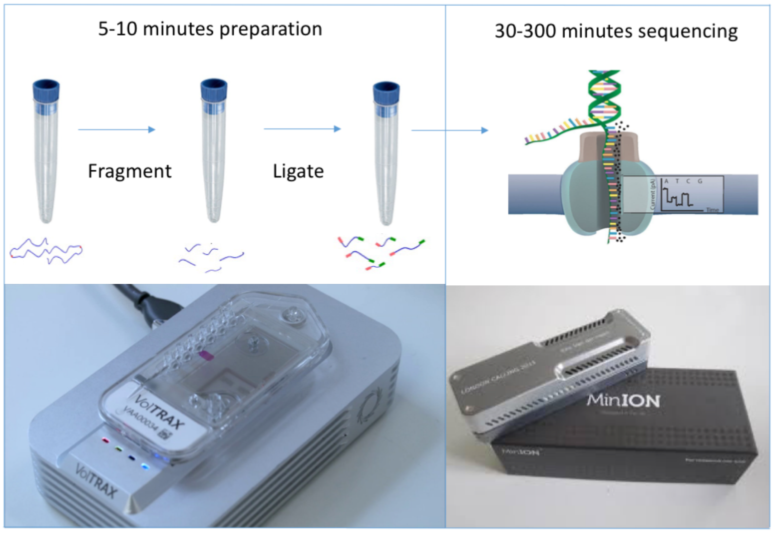


Figure 2: The sample preparation process for sequencing, which consists in the DNA/RNA fragmentation and the ligation of the sequences to chemical primers that enable the sequencing reaction. With the development of the technology, this preparation will be performed automatically using the VolTRAX device, before sequencing using the Oxford Nanopore Minion.

tool has been developed relying on the cloud-based Metrichor platform, which is called *Whats in my Pot?* (WIMP) analysis pipeline [30]. The WIMP application classifies microbial species in real time, using a data structure that is pre-built and shared by all the application runs, which relies on a taxonomy classification and a reference database.

This data structure maps all k-mers of length 24 present in the cho- sen database to nodes in the NCBI taxonomy tree. Because of this pre- processing, new sequencing reads can be quickly classied by looking at k-mers, rather than aligning them against the original reference. During the sequencing run, the reads are basecalled by the Metrichor system and classified by the WIMP pipeline against the pre-built data structure [31].

As introduced, such approach has the drawback of requiring large bandwidth for data transmission, which represents a major drawback considering that this device has been designed to work on the field, also in remote regions. This is the reason why a Fog computing approach well suits such technology. In particular, our approach aims at performing on SoC devices, co-located with the sequencer, the work done on the Cloud by Metrichor and WIMP. This entails in the need of uploading on the Cloud only the metagenomic results, which are basically a few KB containing the identified bacteria and their relative abundance.

3. The SoC Devices

SoC hardware platforms are integrated circuits, designed for the mobile and embedded markets, composed of low power multicore processors combined with a GPU and all the circuitry needed for several I/O devices. In a laboratory hosted at INFN-CNAF data center in Bologna, the COSA project (COmputing On SoC architecture, www.cosa-project.it) built and maintains several racks of low power SoCs, which are available to explore the possibility of a greener, cost-effective and less power hungry scientific computing [32, 33]. In this work, we considered only x86-based hardware, because porting applications to these platforms is straightforward compared to other ones, i.e., ARM based, having all the dependencies already compiled and available [34, 35, 36].

The COSA x86_64 bit cluster includes four mini-ITX boards powered by the Intel C-2750 Avoton SoC, four mini-ITX motherboard based on the Intel Xeon D-1540 CPU, two mini-ITX boards based on the Intel Pentium N3700 processor, two mini-ITX boards based on the Intel Pentium J4205 processor,

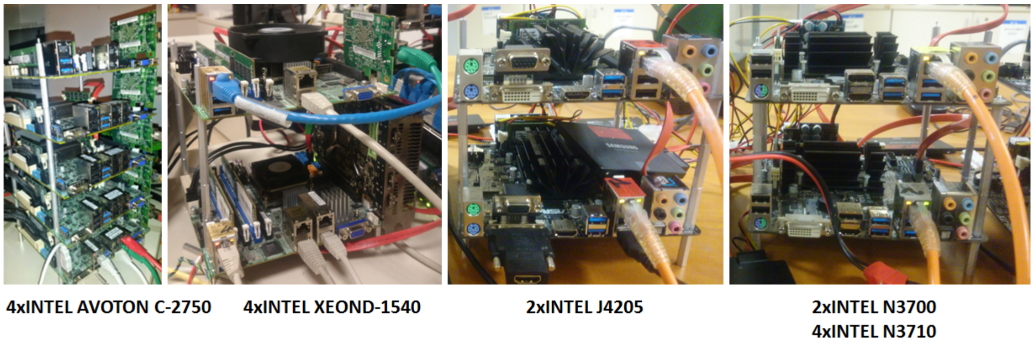


Figure 3: The low power devices used in this work. From left to right: Intel Avoton C2750, Intel XeonD 1540, Intel Pentium J4205, Intel Pentium N3700.

and finally four mini-ITX boards based on the newer Intel Pentium N3710 processor. The remarkably low Thermal Design Power (TDP) of the boards, when declared, ranges from 6W of Intel Pentium N3700 to 45W of the 8-cores Intel Xeon-D processor.

All SoCs are interconnected with standard 1 Gbit/s Ethernet, whereas the Avoton and the XeonD are connected with both 1 Gbit/s and 10 Gbit/s Ethernet connections. Wireless and 4G connections are also available.

A master server is used as a monitoring station and a power meter is attached to the monitoring station to measure the energy consumption for every test or real-life application. On all platforms, the operative system Ubuntu 16.04.2 is installed. An external network file system hosting all software and datasets is mounted on every cluster node.

4. The FoG Infrastructure for Metagenomic Analysis

In this section we describe the FoG infrastructure that relies on SoC devices, for local processing of raw sequencing data and uploading the results on a IoT platform, such as Amazon Web Services (AWS).

In particular, the Fog layer takes in input the sequencing information in terms of variations in the ionic currents of the Nanopore device, which are data in FAST5 format (a variant of HDF5 files). These files are very large, since changes in the ionic currents are measured many times for each base while the DNA/RNA molecules pass through the nanopores. So, FAST5 files in input are more than 10 times larger than the output FASTA sequences. After that, each FASTA file is processed to identify bacteria, which produces in output a tuple as the following:

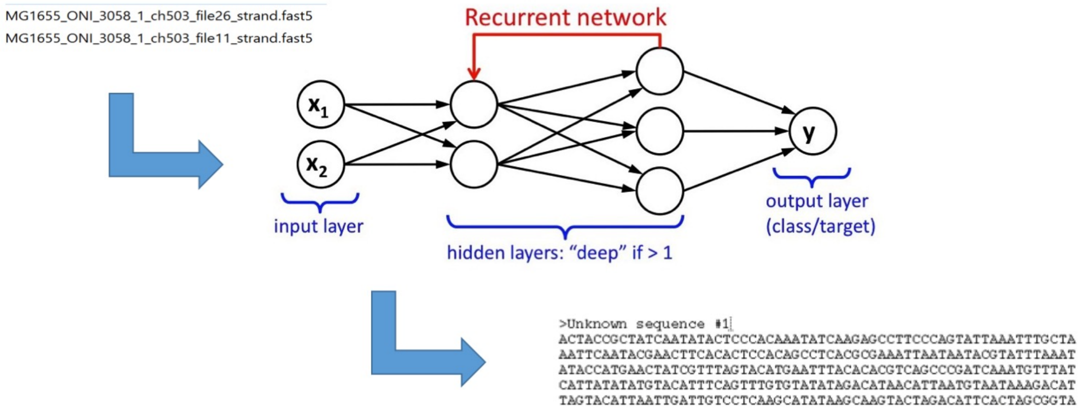
- Percentage of reads covered by the clade rooted at this taxon;
- Number of reads covered by the clade rooted at this taxon;
- Number of reads assigned directly to this taxon;
- A rank code, indicating (U)nclassified, (D)omain, (K)ingdom, (P)hylum, (C)lass, (O)rder, (F)amily, (G)enus, or (S)pecies;
- NCBI taxonomy ID;
- indented scientific name.

This simple output is the only piece of information produced by the Fog layer after processing the results of the analysis performed by MinION. Therefore, only these data will be uploaded to the Cloud infrastructure for data integration and decision making.

4.1. The Bioinformatics pipeline

A few open source base callers have been developed in academia to compete with the proprietary implementation of Metrichor, such as Nanocall [37] and DeepNano [38]. According to our analysis, DeepNano is newer, deals with the last improvements of the Oxford Nanopore technology (chemistry version R9) and also provides better results in terms of accuracy. The reason is that DeepNano relies on deep recurrent neural networks (Figure 4), which are able to capture long-range dependencies, while the HMMs used by Nanocall are only able to model short range dependencies. This results in a significantly improvement of the base calling accuracy achieved by DeepNano on data from chemistry version R7.3, also compared to the default base caller supplied by the manufacturer. On chemistry version R9, DeepNano achieves results comparable to Metrichor [39].

The DeepNano recurrent neural network takes as input a vector consisting of the mean, standard deviation and length of the signal for each detection event and produces as output a vector describing the probability distribution for each called base. A schema of the process is shown in Figure 4. As an optimization algorithm, DeepNano uses a stochastic gradient descent approach, combined with Nesterov momentum [40], to increase the convergence rate.



```
MG1655_ONI_3058_1_ch503_file26_strand.fast5
MG1655_ONI_3058_1_ch503_file11_strand.fast5
```

```
>Unknown sequence #1
ACTACCGCTATCAATATACTCCCACAAATATCAAGAGCCTTCCCAGTATTAAATTTGCTA
AATTCAATACGAACTTCACACTCCACAGCCTCACGCGAAATTAATAATACGTATTTAAAT
ATACCATGAACTATCGTTTAGTACATGAATTTACACACGTCAGCCCGATCAAATGTTTAT
CATTATATATGTACATTTCAGTTTGTGTATATAGACATAACATTAATGTAATAAAGACAT
TAGTACATTAATTGATTGTCCTCAAGCATATAAGCAAGTACTAGACATTCACTAGCGGTA
```

Figure 4: The analysis steps performed by DeepNano, which takes as input files in FAST5 format. This information is analysed through a recurrent neural network, which provides as output a probability distribution of the bases to call.

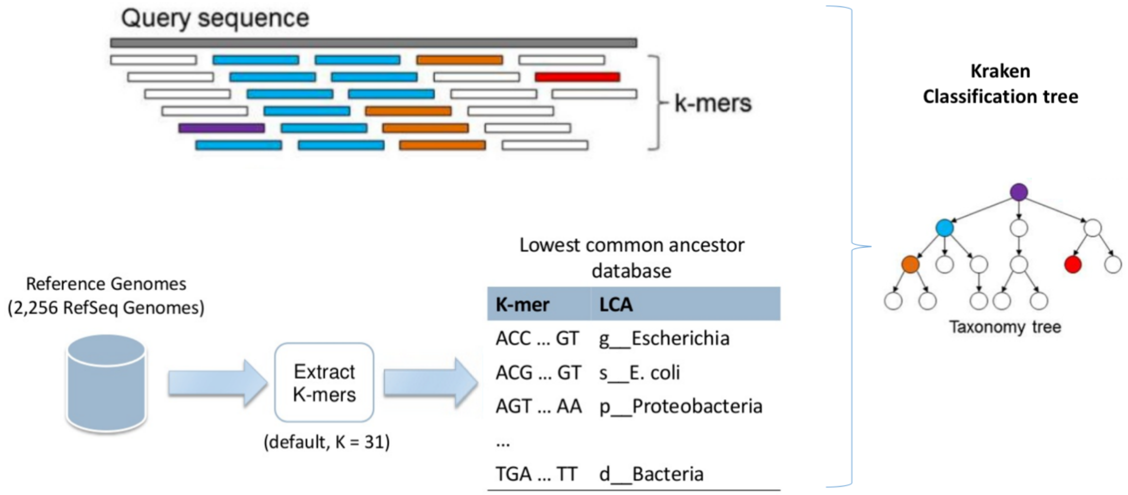


Figure 5: From the 2256 RefSeq Genomes available through NCBI, a database of k-mers (defaulf K=31) is extracted, identifying the lowest common ancestor for each sequence (see text for more details). Kraken works by segmenting the input sequences in k-mers in order to identify these fragments in the lowest common ancestor database. From this analysis a taxonomy tree is identified, which describes the composition of the sample under analysis.

After base calling, we chose Kraken [41] for the identification and quantification of bacteria inside the samples. Kraken is an ultrafast, open source and highly accurate program for assigning taxonomic labels to metagenomic DNA sequences. At the core of Kraken is a database that contains millions of records each one consisting of a k-mer annotated with the lowest common ancestor of all organisms whose genomes contain that k-mer. This database, built using a user-specified library of genomes, allows a quick lookup of the most specific node in the taxonomic tree that can be associated with a given k-mer. Each read is classified by querying the database for each k-mer in the sequence and then using the resulting set of lowest common ancestor taxa to determine an appropriate label for the read. A schema of the process is shown in Figure 5.

4.2. Parallel implementation

MinION streams data as a set of files, that can be processed in an independent way by DeepNano as soon as a new file is available. After base calling with DeepNano, the identification of which bacteria are present in the sample and their amount is done with Kraken. Since the operating manual of the latter software specifies the requirement of a considerable amount of free memory (at least 75 GB) to hold the database in the RAM and run efficiently, we had to download a reduced standard database, called MiniKraken, which is supplied specifically for low-memory computing environments. Indeed, we notice that it is always possible to generate a reduced version of the database considering only relevant k-mers in a sequence to get the correct classification, thus not invalidating the analysis.

Both DeepNano and (Mini)Kraken are sequential software, but it is possible to analyse files produced by MinION in a completely data parallel way. For this reason we implemented a simple application in Python which manages the execution of independent instances of the pipeline DeepNano + Kraken, one for file (or set of files grouped in different folders). The application runs a number of analyses at a time equal to the number of the available cores of the SoC, because we experimentally determined that the use of hyper-threading results in lower performance figures. The software we implemented is available at http://goo.gl/guk7q5. This solution is simpler compared to the installation of a scheduler for managing a single SoC and provides good performance figures. After the end of each analysis, the results are uploaded to the IoT platform, as described below.

4.3. AWS IoT

The IoT platform represents the link to transfer information from the local part of the system, where data are acquired and pre-processed, to the centralized part, where they are stored and analysed. Such a platform can be defined as a software suite that monitors, manages and controls various types of endpoints via applications built upon it [42].

In this work we considered the use of AWS IoT, because its use is straightforward, as we are going to describe, and taking into account the rich integrated environment it enables, as the machine learning platform services [43]. AWS IoT is a cloud managed platform providing a publish/subscribe brokering service. It offers out of the box a number of features as security and the seamless integration with the AWS service ecosystems, like Lambda functions, DynamoDB, S3 and many more.

The platform basically enables the bi-directional communication between Internet-connected *things* (e.g., sensors and applications) through logical channels. The communication is based on JSON messages addressing topics like *sensor/rain/location_xy*. A message broker sends the message to all clients that have registered to receive messages for a topic. The act of sending the message is referred to as *publishing*. The act of registering to receive messages for a topic filter is referred to as *subscribing*.

The general architecture of the platform is depicted in Figure 6. Hereafter a brief description of the main components.

Things and Messages

An entity interacting with the AWS IoT platform corresponds to a *thing* and it is uniquely identified by an Amazon Resource Name. It is possible to define thing types to simplify the management of homogeneous devices. For example they can share a common set of attributes, which describe their identity and capabilities.

Communications between *things* and AWS IoT use the TCP protocol on port 8883 and are protected with of X.509 certificates. Such certificates can be generated by the platform or provided by the user. Moreover, it is necessary to define a policy to authorize each thing/thing type to perform AWS IoT operations, such as subscribing and/or publishing messages. It is possible to revoke a certificate in any moment, in order to block the access for a specific device. Therefore, the authentication/authorization system can address in a flexible and effective way the security problem normally associated with IoT. The resulting of the registration of a new thing is shown in Figure 7.

The platform provides also a pre-configured software connection kit of a few MBs in Java or Python, called NodeJS, that immediately allows to exploit the device to publish/subscribe messages. Many other SDKs are available in order to allow different devices to exploit the capability of the AWS IoT platform, as Javascript, embedded C, C++, C#, Java, PHP, iOS and Android. For example, the NodeJS code to publish messages is the following:

```
var device = awsIot.device({
   keyPath: __dirname+'/myprivate.key',
  certPath: __dirname+'/mycert.pem',
    caPath: __dirname+'/myroot-CA.crt',
```

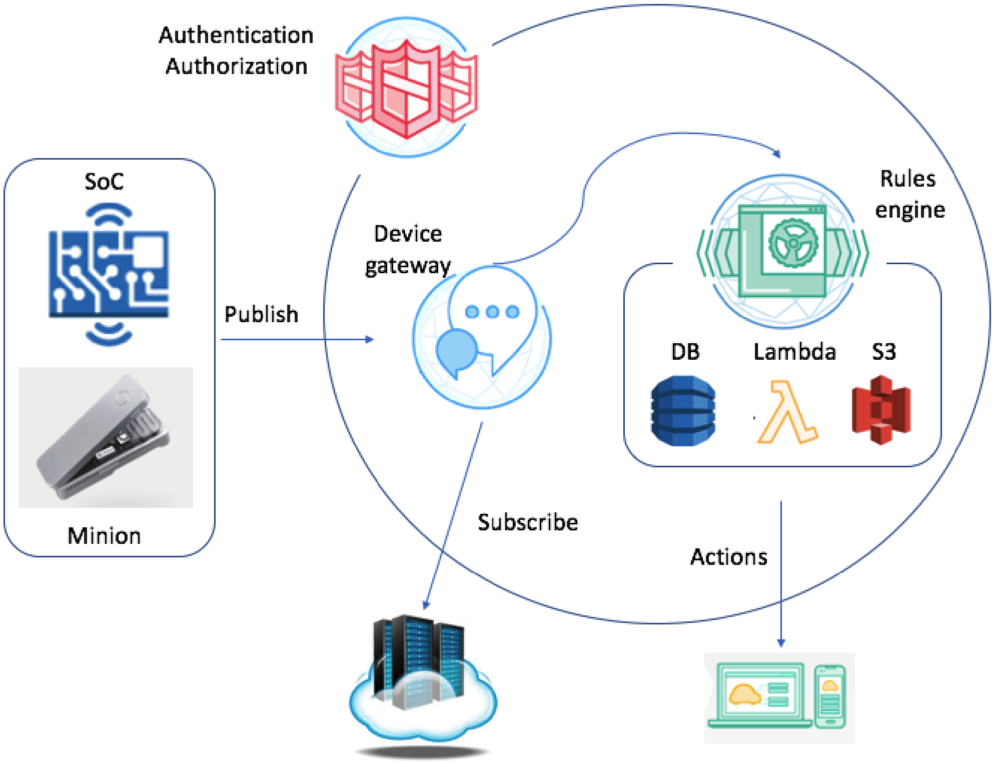


Figure 6: The IoT platform collects information from the Fog layer, enforcing an authentication and authorization policy. Once data reach the device gateway they can be processed through a rules engine or more complex machine learning approaches. Then feedback messages are provided to all subscribers and action are performed according to the defined rules.

```
  clientId: 'Nanopore_2',
      host: 'xxx.iot.us-east-2.amazonaws.com'});

device.on('connect', function() {
  console.log('connect');
  device.subscribe('air_filter/nanopore_2');
  device.publish('air_filter/nanopore_2',
  JSON.stringify({"kind": "C", "bp": 36, "fasta_id": "SEQ1",
  "kraken_id": 562, "LCA_mapping": "562:6" }));
});
```

The result can be inspected through an integrated client, as shown in Figure 7. Our application creates a new NodeJS file, based on the above code containing one *publish* operation per bacteria identified during each single analysis by Kraken.

The Device Gateway

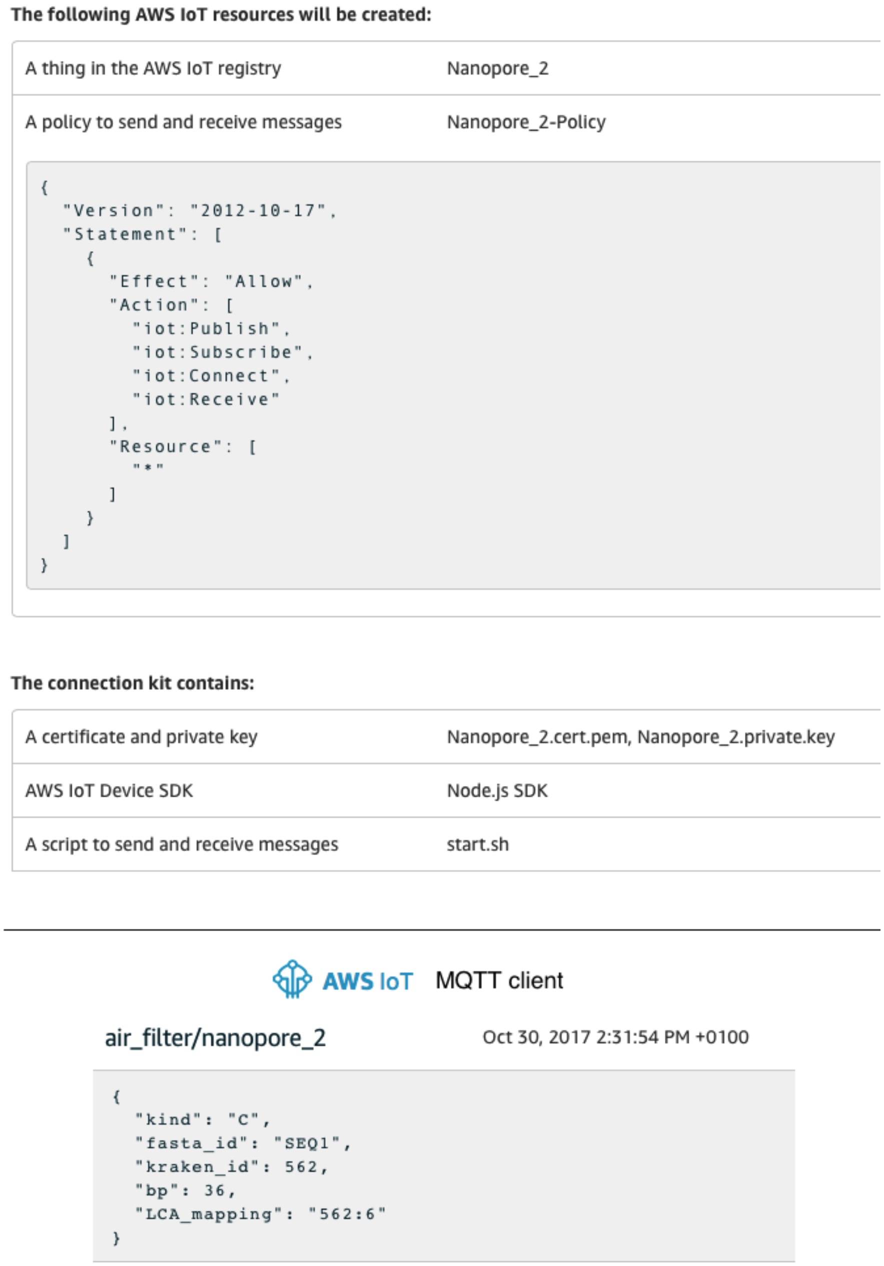
The message broker is called Device Gateway. It implements the publisher/subscriber behaviour using the Message Queue Telemetry Transport (MQTT) protocol. When a message is published on a topic, the device gateway checks for currently connected *things* with a subscription that map to the topic of the message. The broker then forwards the messages to all of them.

Each topic namespace is isolated for each pair (AWS account, region). For example, the (air_filter/nanopore_2, EU (ireland)) topic for an AWS account is independent from any other topic or region. It is not possible sending and receiving messages across accounts and regions.

The simplest architecture that can be implemented with AWS IoT is composed only by *things* and the device gateway. In fact, an external Web Service can register to the device gateway, subscribe to all the topics defined in the applicative scenario and manage them in an independent way, i.e., for storing them on a proprietary repository, for taking actions as sending alerts, or for monitoring applications. Otherwise the rules engine has to be activated.

The Rules Engine

The Rules Engine provides users with the ability to process the incoming messages for automatically performing operations. It is based on rules with



The following AWS IoT resources will be created:

A thing in the AWS IoT registry	Nanopore_2
A policy to send and receive messages	Nanopore_2-Policy

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "iot:Publish",
        "iot:Subscribe",
        "iot:Connect",
        "iot:Receive"
      ],
      "Resource": [
        "*"
      ]
    }
  ]
}
```

The connection kit contains:

A certificate and private key	Nanopore_2.cert.pem, Nanopore_2.private.key
AWS IoT Device SDK	Node.js SDK
A script to send and receive messages	start.sh

AWS IoT MQTT client

air_filter/nanopore_2 Oct 30, 2017 2:31:54 PM +0100

```
{
  "kind": "C",
  "fasta_id": "SEQ1",
  "kraken_id": 562,
  "bp": 36,
  "LCA_mapping": "562:6"
}
```

Figure 7: The AWS implementation. In the upper part, the authentication and authorization setup for a thing using the AWS IoT platform. Below, an example of message reads through the integrated subscription client.

a syntax very similar to an SQL query: it is possible to select only some attributes of the payload, like an SQL projection; it is possible to choose the topic from which filter the messages (with supports also for filtering all messages with a topic that starts with a string); finally, it is possible to specify one or more conditions, as the where clause of an SQL query.

For example, the following rule:

```
select * from 'air_filter/#' where kraken_id = 470
```

will select all messages where Acinetobacter baumannii has been found [6].

The resulting messages can than be processed using a set of ready-to-use actions, such as:

- insert a message into a DynamoDB table;
- send messages to an Amazon Kinesis Stream;
- republish messages to (a different) AWS IoT Topic;
- invoke a Lambda function passing the message data;
- store messages to an S3 bucket.

The difference of having a web service subscribing a topic or the use of an action relies on the scalability AWS can provide. In the first case, the web service has to process every message with the computational capabilities of the server where it runs, extract only the relevant messages and then it can takes proper actions. In the second case, a rule can be defined in order to filter the incoming messages as the one above. The filtered messages, for example, can be re-published with a new user-defined topic, e.g., *alert*. Therefore, the web service can subscribe only the *alert* topic, levering the heavy computational part to the AWS capabilities.

The same applies to the storing of messages as time series on AWS operated databases or Simple Storage Service (S3) instances. Lambda functions instead represent a piece of code triggered by messages of a topic. The user can process the data with her/his local web service and interact with further AWS or external web services and databases.

In our current prototype we exploit the Rules Engine to filter the messages, while the actual alarm triggering is managed through a dedicated web services. Also the storing of results was performed with local resources.

5. Experimental Results

Given the algorithmic background detailed in Section 3, we cloned the bitbucket repository of DeepNano (chemistry version R9) on the SoC boards, which requires Python 2.7 and the Theano library [44] for training neural networks, and we downloaded and compiled Kraken, written in C++. The execution of both tools was straightforward, as the use of the AWS IoT SDK based on NodeJS.

Beside assessing the feasibility of the system architecture, we performed some quantitative tests to evaluate the effectiveness, performance and power consumptions of some metagenomic experiments.

5.1. Parallel Performance

We assessed the performance of the SoC platforms of the COSA cluster on data available on public repositories. Unfortunately, only a few datasets can be found concerning metagenomic experiments, because the technology is indeed very new. This also because typically users load their data to the Oxford Nanopore Cloud-based platform and get FASTQ files back, therefore it is difficult to find raw data. Considering these constrains we selected the experiments reported in [46]. In particular, we selected 4 datasets among the available ones, with a different size.

The considered datasets are: ERR1713483 (283,294 bp), ERR1713487 (337,913 bp), ERR1713485 (427,280 bp), and ERR1713486 (517,628 bp), which correspond to metagenomes generated by mixing sequences from different species. These experiments have been chosen in order to test our approach on datasets of different sizes, in order to evaluate the impact of the number of bp on the performance of the pipeline.

Figures 8 and 9 show the number of bases per seconds processed by DeepNano and Kraken, respectively for the considered datasets and for increasing number of processing cores in the different SoCs. It is worth to note that only Avoton and XeonD are equipped with 8 cores, while the other three SoCs has only 4 cores.

As regards DeepNano, XeonD is able to process 1,600 bases per second, Avoton about 1,180, N3700 about 580 while J4205 and N3710 only 200. Higher values - in the order of million bases per second - have been achieved with Kraken, where the difference between the slowest and the fastest SoC are lower than with DeepNano.

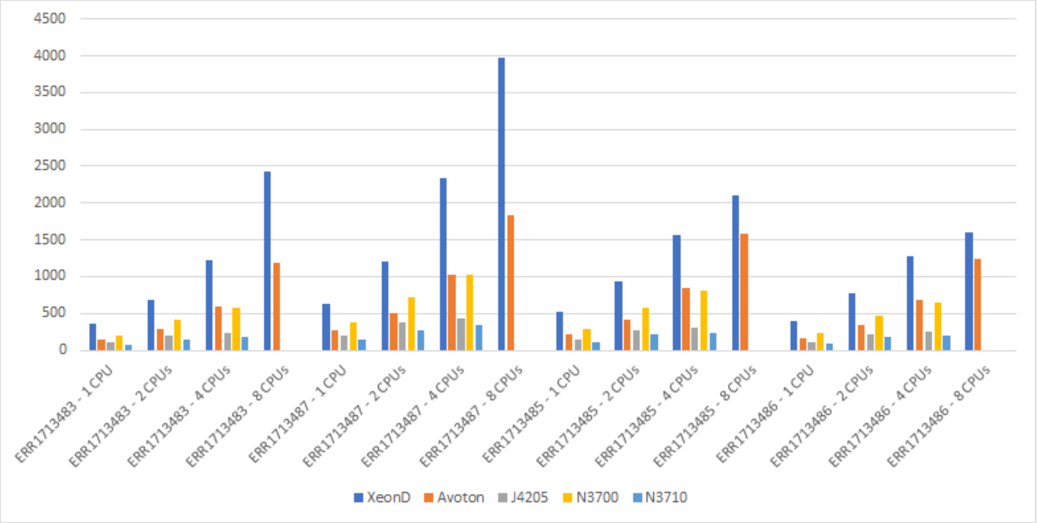


Figure 8: Bases per seconds processed by DeepNano for different datasets and for increasing number of processing cores (1, 2, 4, 8 in the x-axis label) in the different SoCs (colour coded).

The most interesting result has been achieved with DeepNano for the ERR1713487 dataset, which computations scales much better than the others. This can be probably explained with a better quality (less errors) of the sequences in the datasets, which reduces the number of iterations needed by the recurrent neural network to converge.

Moreover, Figures 10 and 11 show the scaling of the considered applications for the different datasets using the widely adopted metric of speedup.

Apart from the Avoton SoC, for which the scaling with the number of cores is very good, the other platforms exhibit a poor scaling, which might be ascribable to the high usage of the RAM by DeepNano and Kraken.

To support this hypothesis, we run standard STREAM synthetic benchmarks to estimate how quickly data can be written to or read from memory by the processor for simple vector kernels (here we show *scale*) in all the SoCs considered here. Results are shown in Figures 12, where the memory bandwidth is plotted as a function of vector size (left) and as a function of the number of CPU cores for a fixed array size of 50 MB (right). We can see that only Avoton does not present lower performance by using all the available cores. XeonD instead decreases the performance by using 8 cores with respect to 4 and the other three SoCs present the same behaviour us-

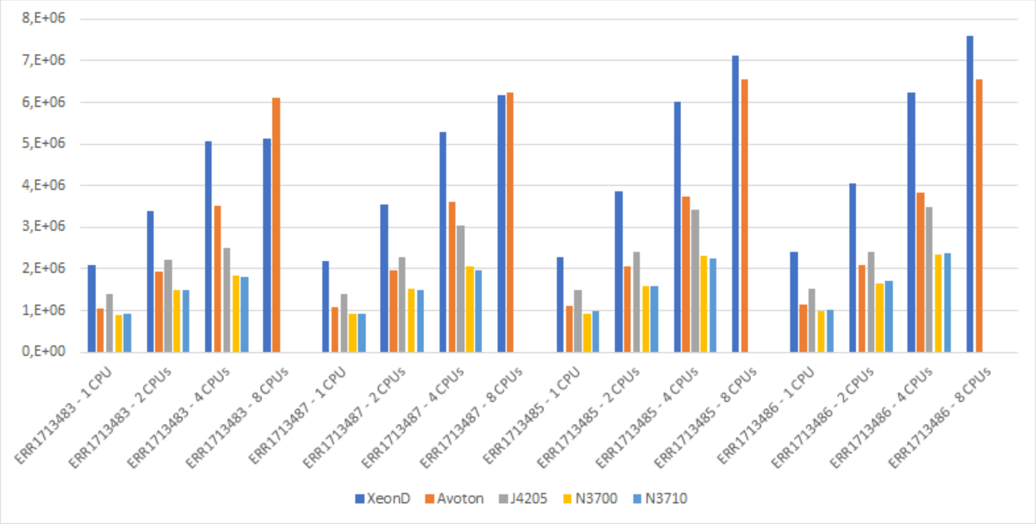


Figure 9: Bases per seconds processed by Kraken for different datasets and for increasing number of processing cores (1, 2, 4, 8 in the x-axis label) in the different SoCs (colour coded).

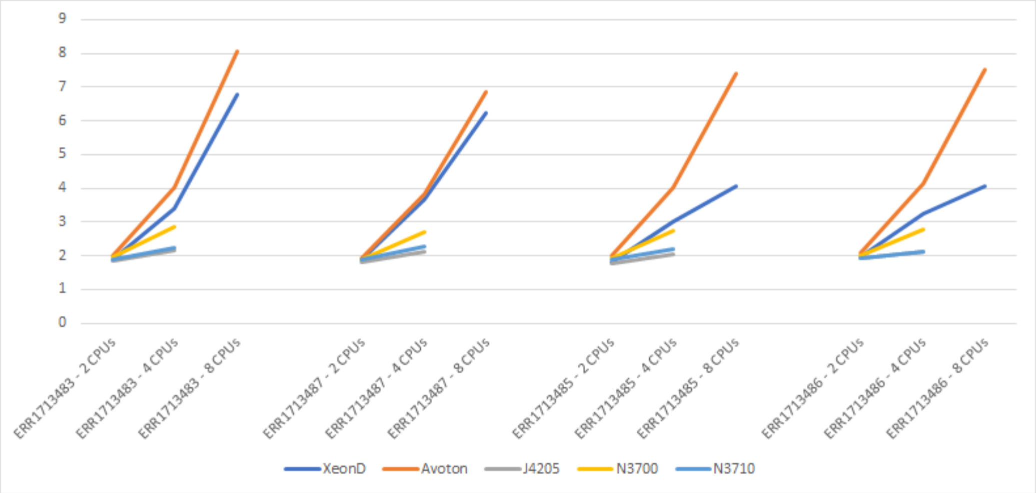


Figure 10: Speedup of DeepNano for different datasets and for the different SoCs (colour coded).

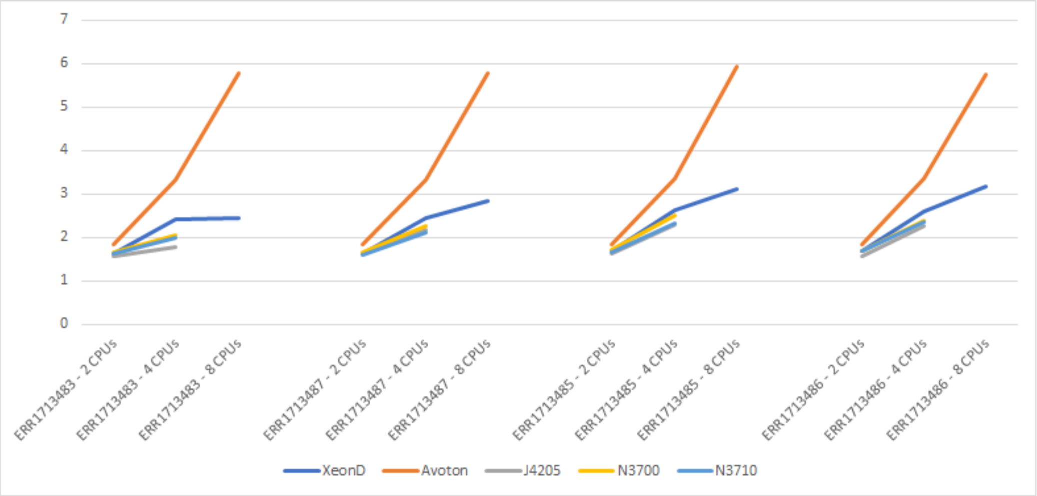


Figure 11: Speedup of Kraken for different datasets and for the different SoCs (colour coded).

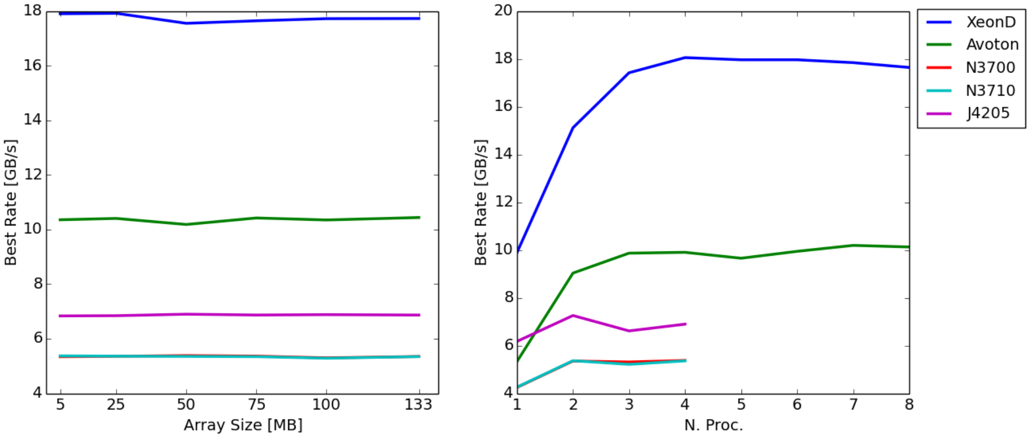


Figure 12: STREAM benchmark: sustainable memory bandwidth (GB/s) for the SoCs. Left plot: for increasing size of the array, STREAM is executed for the maximum number of available cores in the platform. Right plot: for fixed array size (50 MB), increasing number of cores in the platforms (8 cores for XeondD and Avoton, 4 cores for the others).

ing 4 cores instead than 2. Moreover, the sustainable memory bandwidth of N3700 and N3710 is indeed remarkably low, when compared to that of the other SoCs.

Furthermore, it is not completely clear why the newer Pentium J4205 performs worse than the older N3700 with DeepNano. The same holds true with N3710, that has an higher frequency. This aspect is currently under investigation, particularly for what concerns the management of the Turbo frequency. A different handling of the Turbo frequency could also be responsible for the Avoton SoC scaling better than the XeonD SoC.

In any case the following considerations hold true:

- the use of all the available cores for running independent analyses results in an increase of performance with respect to the use of less cores;
- the execution time of DeepNano is higher than Kraken: the former processes about one thousand bases per second and the latter more than one million;
- Avoton presents the best scalability results, but XeonD provides the highest performance most of the times.

5.2. Power Consumption and Data Size in a Real-World Scenario

The focus of the previous Section was to assess the basecalling speeds of the low-power SoCs of the COSA cluster, which represent the key components of the Fog layer. Here we consider also the power consumption and, even more important, the actual size of the data to be sent to AWS IoT in a real-world scenario.

A full discovery metagenomic experiment can produce up to 2 Gbp per day of MinION sequencing, resulting in an equivalent amount of GByte (1 base pair - 1 byte). When the experiment performed by MinION has a monitoring purpose, it is common practice to schedule one analysis per day. For monitoring purpose this accuracy is usually not necessary, since we want only to identify the presence of few specific strains. Therefore, the resulting dataset has a size of about 100 Mbp, corresponding to about 30 minutes of Minion sequencing.

As dataset for our tests we considered a set of 290 files, with a global size of 100 MB. They are processed by DeepNano that produces the same amount of files, but with a global size of 4.6 MB in the FASTA format, further processed by Kraken. The final result is represented by 790 tuples having

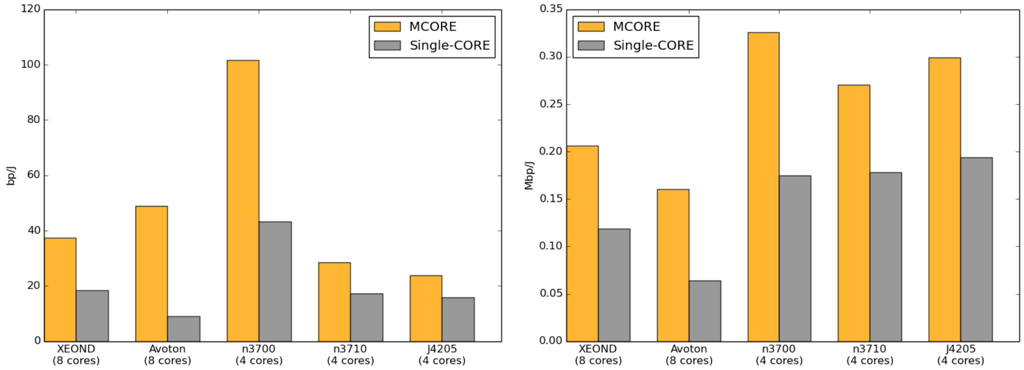


Figure 13: Bases processed per Joule by DeepNano (left) and by Kraken (right) for the different low-power architectures considered in this work. Grey histograms refer to the single-core runs, whereas yellow histograms show performances when all the available cores in the machines are used by parallel instances of the code.

the format described in Section 4, with a global size of 3.2 MB. Therefore the proposed system is able to reduce to 1/30 the data stream size to be sent over the Internet connection subdivided in 790 messages.

Figure 13 (left plot) shows the energy performance of such workload on low-power boards, i.e., the number of basecalls processed per Joule, while Figure 13 (right plot) shows the number of bases processed by Kraken per Joule to perform the bacteria identification.

We can see that N3700 presents the best result. Surprisingly, the most most powerful SoCs are not the most power hungry for DeepNano, while this is true for Kraken.

5.3. Discussion

As stated before, the proposed system is a feasible solution to manage the data stream produced by MinION devices, because on average it reduces to 1/30 the data stream size to be sent over the Internet connection subdivided in about 800 messages.

As regards the execution times, the measurements described above indicate the following capacities with DeepNano for each board:

- XeonD can process around 157 Mbp/day using 4.2 MJ,
- Avoton around 106 Mbp/day using 2.2 MJ,

- N3700 around 53 Mbp/day using 0.5 MJ,
- N3710 and J4205 around 20 Mbp/day using 0.8 MJ.

For all the considered SoCs, instead, Kraken is able to process hundreds of Gbp per day with little energy consumption. Therefore, the bottleneck of the system is represented by the DeepNano application. Only devices equipped with a single XeonD or Avoton are able to process an analysis per day, while for the other tested platforms there is the need to include 2-5 boards to get results in time. To provide a comparison figure, the Xeon E5-2683 v3 CPU (14 Cores, 28 Threads, 2.0GHz, 3.0GHz Turbo, 35MB Cache, 120W), used in a High Performance Computing (HPC) cluster representing the reference for the COSA project, is able to process about 1,000 bp per second with a single core with respect to the 400 of the XeonD. Therefore, using all its 14 physical cores, it is able to scale up to 1 Gbp/day, which means that there is the need to use at least two Xeon E5 cpus for processing the theoretical 2 Gbp produced by a full discovery metagenomic experiment with MinION.

In a realistic scenario, MinION produces 30 bp per second per pore: considering 512 pores, the data stream is of 15,360 bp per second, i.e., 55 Mbp per hour, i.e., 2.64 Gbp in 48h [47]. Therefore, considering the best-performing SoCs, i.e., the XeonD and Avoton, 9 and 13 boards respectively are needed in order to achieve real time computations. Many more boards would be required for real-time computations using different SoCs, e.g. J4205.

However, while its computing performances are overall superior, the XeonD SoC is by far the most power-hungry of the considered SoCs. This means that it cannot be battery-powered (although it can be probably powered by solar panels) and a box containing 9 mini-ITX motherboards (15 cm size each) would resemble the size of a shoebox. This approach can be used in scenarios where timeliness is extremely important.

If time is not a problem, computations can be performed using a flash storage to temporary store data. In this scenario a single XeonD or Avoton SoC can support the throughput of a single day analysis and the whole system can be battery-powered. This configuration is more than enough for monitoring the bacterial composition of cultivated soils and animal samples for keeping a set point in the microbioma composition.

On the other hand, the less power-hungry architectures represent a suitable solution when real-time computations are not mandatory, since the other SoCs, and in particular the Pentium processors, are available on the market in remarkably smaller form factors, e.g., as small as credit cards, and can be battery powered.

6. Conclusions and Future Directions

Metagenomic studies are becoming increasingly widespread, yielding important insights into microbial communities covering diverse environments from terrestrial and aquatic ecosystems. With the advent of high-throughput sequencing platforms, the use of large scale shotgun sequencing approaches is now commonplace.

While portable sequencing machine can be used for monitoring bacterial communities on the field, also in remote regions of the world, their actual exploitation is partially impaired by the need of *always-on* network connections for heavy data streaming. We presented a system based on the Fog computing paradigm representing an effective solution to fully exploit the huge potentiality of these devices.

In particular, experimental results show how System-on-Chip low-power devices can be used to analyse the data stream of the Oxford Nanopore Minion, both in real-time or in predefined time intervals, accordingly to the requirements of the considered scenario, by using local storage capabilities. Our experience discusses also energy consumption requirements.

We plan to extend this work to manage a dynamic scenarios. For example, if dangerous bacteria are identified in an air filter, the analysis can be performed more frequently. In this case, we will analyse the trade-off between putting more computational capabilities on the local device and transferring more data to the Cloud infrastructure.

Acknowledgement

The Minion Oxford Nanopore technology was available to the project thanks to Pietro Lio, University of Cambridge, who was involved in the first Oxford Nanopore MinION Access Program. The accessibility to the AWT platform was granted by a AWS Cloud Credits for Research to Ivan Merelli. This work has been supported by the Italian flagship initiative Interomics for CNR researchers and by the INFN COSA Project for CNAF-INFN researchers.

References

[1] Maksimovic M., Vujovic V. (2017) Internet of Things Based E-health Systems: Ideas, Expectations and Concerns. In: Khan S., Zomaya A., Abbas A. (eds) Handbook of Large-Scale Distributed Computing in Smart Healthcare. Scalable Computing and Communications. Springer, Cham

[2] Flavio Bonomi et al., Fog computing and its role in the internet of things. In Proceedings of the first edition of the MCC workshop on Mobile cloud computing. ACM, 2012.

[3] Amir Vahid Dastjerdi and Rajkumar Buyya. 2016. Fog Computing: Helping the Internet of Things Realize Its Potential. Computer 49, 8 (August 2016), 112-116.

[4] Fog Computing and the Internet of Things: Extend the Cloud to Where the Things Are. CISCO Whitepaper, 2015

[5] Jain, M., Olsen, H. E., Paten, B., et al (2016). The Oxford Nanopore MinION: delivery of nanopore sequencing to the genomics community. Genome Biology, 17(1), 239.

[6] Alsan, Marcella, and Michael Klompas. Acinetobacter Baumannii: An Emerging and Important Pathogen. Journal of clinical outcomes management: JCOM 17.8 (2010): 363369. Print.

[7] Peleg, Anton Y., and David C. Hooper. Hospital-Acquired Infections Due to Gram-Negative Bacteria. The New England journal of medicine 362.19 (2010): 18041813. PMC. Web. 20 Oct. 2017.

[8] Tan, BoonFei et al. Next-Generation Sequencing (NGS) for Assessment of Microbial Water Quality: Current Progress, Challenges, and Future Opportunities. Frontiers in Microbiology 6 (2015): 1027. PMC. Web. 20 Oct. 2017.

[9] Daniel Rolf. The metagenomics of soil. Nature Reviews Microbiology, 2005, 3.6.

[10] Castaeda, Luis E., and Olga Barbosa. Metagenomic Analysis Exploring Taxonomic and Functional Diversity of Soil Microbial Communities in

Chilean Vineyards and Surrounding Native Forests. Ed. Keith Crandall. PeerJ 5 (2017): e3098. PMC. Web. 20 Oct. 2017.

[11] GANDA, Erika K., et al. Longitudinal metagenomic profiling of bovine milk to assess the impact of intramammary treatment using a third-generation cephalosporin. Scientific reports, 2016, 6.

[12] Arwyn Edwards, Aliyah R Debbonaire, Birgit Sattler, Luis AJ Mur, Andrew J Hodson, Extreme metagenomics using nanopore DNA sequencing: a field report from Svalbard, 78 N Biorxiv, 2016.

[13] Sarah L Castro-Wallace, Charles Y Chiu, Kristen K John, Sarah E Stahl, Kathleen H Rubins, Alexa B. R. McIntyre, Jason P Dworkin, Mark L Lupisella, David J Smith, Douglas J Botkin, Timothy A Stephenson, Sissel Juul, Daniel J Turner, Fernando Izquierdo, Scot Federman, Doug Stryke, Sneha Somasekar, Noah Alexander, Guixia Yu, Christopher Mason, Aaron S Burton, Nanopore DNA Sequencing and Genome Assembly on the International Space Station. Biorxiv, 2016.

[14] Alexa B R McIntyre, Lindsay Rizzardi, Angela M Yu, Noah Alexander, Gail L Rosen, Douglas J Botkin, Sarah E Stahl, Kristen K John, Sarah L Castro-Wallace, Ken McGrath, Aaron S Burton, Andrew P Feinberg, Christopher E Mason, Nanopore Sequencing in Microgravity. npj Microgravity 2, Article number: 16035, 2016

[15] Alexander L. Greninger, Samia N. Naccache, Scot Federman, Guixia Yu, Placide Mbala, Vanessa Bres, Doug Stryke, Jerome Bouquet, Sneha Somasekar, Jeffrey M. Linnen, Roger Dodd, Prime Mulembakani, Bradley S. Schneider, Jean-Jacques Muyembe-Tamfum, Susan L. Stramer and Charles Y. ChiuRapid metagenomic identification of viral pathogens in clinical samples by real-time nanopore sequencing analysis. Genome Medicine20157:99.

[16] G.D. Guerrero, J.M. Cebrin, H. Prez-Snchez, J.M. Garca, M. Ujaldn, J.M. Cecilia, (2013) Towards energy efficiency heterogenous processors: Findings on Virtual Screening methods, Concurrency and Computation: Practice and Experience, 26(10), 1832-1846. doi: 10.1002/cpe.3119.

[17] G.D. Guerrero, R.M. Wallace, J.L. Vazquez-Poletti, J.M. Cecilia, J.M. Garca, D. Mozos, H. Prez-Snchez, (2013) A Performance/Cost Model for

a CUDA Drug Discovery Applica- tion on Physical and Public Cloud Infrastructures, Concurrency and Computation: Practice and Experience, 26(10), 1787-1798. doi: 10.1002/cpe.3117.

[18] Todri-Sanial, Aida, and Sanjukta Bhanja. "Special Issue on Advances in Design of Ultra-Low Power Circuits and Systems in Emerging Technologies." ACM Journal on Emerging Technologies in Computing Systems 12, no. 2 (2015): 11.

[19] Dai L1, Gao X, Guo Y, Xiao J, Zhang Z. Bioinformatics clouds for big data manipulation. Biol Direct. 2012 Nov 28;7:43;

[20] Konstantinos Krampis, Tim Booth, Brad Chapman, Bela Tiwari, Mesude Bicak, Dawn Field, and Karen E Nelson. Cloud BioLinux: preconfigured and on-demand bioinformatics computing for the genomics community. BMC Bioinformatics. 2012; 13: 42.

[21] I Merelli, H Perez-Sanchez, S Gesing, D DAgostino Managing, analysing, and integrating big data in medical bioinformatics: open problems and future perspectives. BioMed research international 2014

[22] Afgan E, Chapman B, Jadan M, Franke V, Taylor J. Using cloud computing infrastructure with CloudBioLinux, CloudMan, and Galaxy. Curr Protoc Bioinformatics. 2012 Jun;Chapter 11:Unit11.9

[23] D D'Agostino, A Clematis, A Quarati, D Cesini, F Chiappori, L Milanesi, I. Merelli Cloud infrastructures for in silico drug discovery: economic and practical aspects. BioMed research international 2013.

[24] Amir M. Rahmani, Pasi Liljeberg, Jrgo-Sren Preden, Axel Jantsch Fog Computing in the Internet of Things - Intelligence at the Edge Springer, Cham

[25] Biswas, Arnab Kumar, S. K. Nandy, and Ranjani Narayan. Multiprocessor system-on-chip for processing data in cloud computing. Data Security in Cloud Computing (2017): 65.

[26] F. Conti et al., An IoT Endpoint System-on-Chip for Secure and Energy-Efficient Near-Sensor Analytics, in IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 64, no. 9, pp. 2481-2494, Sept. 2017. doi: 10.1109/TCSI.2017.2698019

[27] Yanxiao Feng, Yuechuan Zhang, Cuifeng Ying, Deqiang Wang, Chunlei Du, Nanopore-based Fourth-generation DNA Sequencing Technology, In Genomics, Proteomics & Bioinformatics, Volume 13, Issue 1, 2015, Pages 4-16,

[28] Jain, M., Fiddes, I. T., Miga, K. H., Olsen, H. E., Paten, B., Akeson, M. (2015). Improved data analysis for the MinION nanopore sequencer. Nature methods, 12(4), 351-356.

[29] T. Laver, J. Harrison, P.A. O'Neill, K. Moore, A. Farbos, K. Paszkiewicz, D.J. Studholme, Assessing the performance of the Oxford Nanopore Technologies MinION, In Biomolecular Detection and Quantification, Volume 3, 2015, Pages 1-8

[30] Sissel Juul, Fernando Izquierdo, Adam Hurst, Xiaoguang Dai, Amber Wright, Eugene Kulesha, Roger Pettett, Daniel J Turner, What's in my pot? Real-time species identification on the MinION, biorxiv.

[31] Bonnie L. Brown, Mick Watson, Samuel S. Minot, Maria C. Rivera, Rima B. Franklin MinION nanopore sequencing of environmental metagenomes: a synthetic approach. Gigascience. 2017 Mar; 6(3), pp. 110.

[32] Cesini, D., Corni, E., Falabella, A., Ferraro, A., Morganti, L., Calore, E., Schifano, S.F., Michelotto, M., Alfieri, R., De Pietri, R., Boccali, T., Biagioni, A., Lo Cicero, F., Lonardo, A., Martinelli, M., Paolucci, P.S., Pastorelli, E., Vicini, P.: Power-Efficient Computing: Experiences from the COSA project. In: Scientific programming, volume 2017, Article ID 7206595, 2017.

[33] Daniele D'Agostino, Daniele Cesini, Elena Corni, Andrea Ferraro, Lucia Morganti, Alfonso Quarati, Ivan Merelli: Performance and Economic Evaluations in Adopting Low Power Architectures: A Real Case Analysis. GECON 2017: 177-189

[34] Morganti, L., Cesini, D., Ferraro, A.: Evaluating Systems on Chip through HPC Bioinformatics and Astrophysics Applications. Proceedings of the 24th Euromicro International Conference on Parallel, Distributed, and Network-Based Processing (PDP) 2016, pp. 541-544.

[35] Corni, E., Morganti, L., Morigi, M.P., Brancaccio R., Bettuzzi, M., Levi, G., Peccenini, E., Cesini, D., Ferraro, A.: X-Ray Computed Tomography Applied to Objects of Cultural Heritage: Porting and Testing the Filtered Back-Projection Reconstruction Algorithm on Low Power Systems-on-Chip. In: 24th Euromicro International Conference on Parallel, Distributed, and Network-Based Processing (PDP) 2016, 369-372, 2016.

[36] Morganti, L., Corni, E., Ferraro, A., Cesini, D., DAgostino, D., Merelli, I.: Implementing a Space-Aware Stochastic Simulator on Low-Power Architectures: A Systems Biology Case Study. In: 25th Euromicro International Conference on Parallel, Distributed, and Network-Based Processing (PDP) 2017, 2017.

[37] David, M., Dursi, L.J., Yao, D., Boutros, P.C., Simpson, J.T. Nanocall: an open source basecaller for Oxford Nanopore sequencing data. Bioinformatics. 2017 Jan 1;33(1):49-55.

[38] Boza, V., Brejova, B., Vinar, T. (2017). DeepNano: Deep recurrent neural networks for base calling in MinION nanopore reads. PloS one, 12(6), e0178751.

[39] de Lannoy C, de Ridder D and Risse J. A sequencer coming of age: De novo genome assembly using MinION reads. F1000Research 2017, 6:1083

[40] Sutskever, I., Martens, J., Dahl, G., and Hinton, G. (2013). On the importance of initialization and momentum in deep learning. In Proceedings of the 30th International Conference on Machine Learning (ICML-13), pages 1139-1147.

[41] Wood DE, Salzberg SL (2014) Kraken: ultrafast metagenomic sequence classification using exact alignments, Genome Biology 201415:R46.

[42] Velosa, A., Natis, Y.V., Pezzini, M., Lheureux, B.J., Goodness, E.: Gartners Market Guide for IoT Platforms (2015)

[43] Medvedev, Alexey, Alireza Hassani, Arkady Zaslavsky, Prem Prakash Jayaraman, Maria Indrawan-Santiago, Pari Delir Haghighi, and Sea Ling. "Data Ingestion and Storage Performance of IoT Platforms: Study

of OpenIoT." In International Workshop on Interoperability and Open-Source Solutions, pp. 141-157. Springer, Cham, 2016.

[44] Bergstra, J., Breuleux, O., Bastien, F., Lamblin, P., Pascanu, R., Desjardins, G., Turian, J., Warde-Farley, D., and Bengio, Y. (2010). Theano: a CPU and GPU math expression compiler. In Proceedings of the Python for Scientific Computing Conference (SciPy).

[45] NI, Jiajia; YAN, Qingyun; YU, Yuhe. How much metagenomic sequencing is enough to achieve a given goal?. Scientific reports, 2013, 3.

[46] Brown, B. L., Watson, M., Minot, S. S., Rivera, M. C., Franklin, R. B. (2017). MinION nanopore sequencing of environmental metagenomes: a synthetic approach. GigaScience, 6(3), 1-10.

[47] Lindgreen S, Adair KL, Paul P. Gardner An evaluation of the accuracy and speed of metagenome analysis tools, Scientific Reports 6, Article number: 19233 (2016).