Report - SPSAS on Learning from Data

William Suzuki *

August 2019

This report makes an overview description of the presentations of São Paulo School of Advanced Science (SPSAS) on Learning from Data. SPSAS learning from data took place at the University of São Paulo in the city of São Paulo, from July 29 to August 9, 2019.

Session: Learning From Data

Presenter: Abu-Mostafa, professor at California Institute of Technology, author of *Learning from Data*

The presentation takes a technical and historical overview approach. It is explained the buzzword terminology: Artificial Intelligence (AI), Machine Learning (ML), Big Data, and Neural Networks (NN). Essentially Deep Learning is the same as Neural Networks. It was in the 1980s that the name of the field changed and people started to use Deep Learning. According to the presenter, this was a "brilliant" marketing strategy to create a hype and a sense of a new and flourishing topic of research.

In turn, Neural Networks are a subset of Machine Learning, and in a technical content Machine Learning is approximately the same as Artificial Intelligence. But the terms refer ultimately to different concepts, in the sense that Artificial Intelligence is the goal of Machine Learning. The term Machine Learning is used in contrast to Human Learning. Now it is important to explain what it is meant by intelligence, and what are the characteristics of intelligence.

The goal of the field of Machine Learning is to achieve "intelligence". The important characteristics of an intelligent entity are: i) Perform complex tasks; ii) Learn new skills; iii) Innovation; iv) Taking Over/ Rebellion.

The eminent example is a NN algorithm that plays Go. The game Go is a complex task. The algorithm can adapt to new rules and new environments. The NN model played moves that were unusual and innovative for humans, some strategies that the machine took were not taught to the algorithm.

And the last point is rebellion and the capacity of the machine to take over humans. The risk of humanity becoming second class citizens. Elon Musk and Stephen Hawking are two important and intelligent figures that believe in the dangers of rebellion in Artificial Intelligence. The position of Mustafa is that the problem is not the superhuman machines and algorithms per se but how people use them. He explains that this is a situation much like the case of nuclear weapons. Artificial Intelligence will not take over by its own will, but in the hands of hackers and terrorists, Artificial Intelligence can be very dangerous.

Historical Perspective. From evolution to revolution.

For most of the field of machine learning, researchers were taking the "low hanging fruits", researchers and practitioners were applying ML to known problems, problems which they knew that it could be useful. The following phase was with the very ambitious goals problems, related to superhuman performance in intelligence tasks. And this phase began exactly 7 years ago with the publication of Alex Krizhevsky.

In the 1950s Machine Learning and AI were very simple and rudimentary due to low computer power of the time. There were only simple models being applied, and there was a lot of hype, but the field did not deliver much. In the 1970s AI became a bad word in the private sector because it was becoming known as a waste of resources with few returns. In the 1980s Neural Networks was the new hype, and there was a feeling of great future for the field and great hype. In the 1990s NN was not that good, it did not deliver results. In 2010 people started to say that NN is awesome after all. Nowadays Machine Learning and AI seem to have a great future, and there is another hype.

For Mustafa "hype" is not always a bad word. Where especially in research there is inherent uncertainty where one cannot foresee the results, on the other hand, if we already know the outcome of the

*Department of Economics - FEA-RP - Universidade de São Paulo. - email: william.suzuki@usp.br; github: https://github.com/williamszk; site: williamszk.github.io.

work, it is called development and not research. In research, one can spend 2 or 5 years and do not get the expected results. Hence in the research profession, people are very optimistic so that there is a higher propensity for funding this is how hype can help promote research in machine learning.

The perception about AI change dramatically after the machine could surpass the human capacity, but before this level, people were not impressed by the achievements of the field.

What makes exactly 7 years ago be a turning point for the research in machine learning is that in the fall of 2012 Alex Krizhevsky published a paper called Krizhevsky et al. (2012) where he started making use of (Graphical Processing Units) GPU and applied NN with more layers. With this he achieved a great breakthrough in image recognition and started a rapid expansion in the field, Mustafa and Atlas put great emphasis in this work, claimed to be the greatest contribution in ML.

Now the machines are capable of "innovation" because they can execute actions that we were not expecting. The prominent example is in the game Go and chess. Those are cases where the algorithms started to use moves and strategies never thought by a human being. This is one of the important messages of AlexNet, the algorithm presented at Krizhevsky et al. (2012), is that there are ways to solve problems that we did not foresee.

For great breakthroughs in science, there is usually a lot of building work, to establish the foundations for the breakthrough. For the building phase researcher do not receive much credit, but in the impressive finding, which is the culmination of the building phase, researchers receive credit. This according to Mustafa is what happened in Machine Learning, so it is important to also recognize the foundation work that paved the way for recent leap advancements in the field.

The 1st wave of advances in Machine Learning occurred in finance because it had the best data available. It was used successfully used in forecasting and credit approval algorithms. This was the biggest earlier success, an example of this are investments by financial companies in a firm called Hecht-Nielsen Neurocomputer Corp. The 2nd wave came with e-commerce, also there was a bonanza of collected data. There were developments in algorithms that recommend and make profiling of people. Both contributed to the matching of products and clients. For example the Netflix competition, that occurred between 2006-2009. With this competition, people started to realize that one does not need to have a Ph.D. in computer science to compete and achieve great results. The 3rd wave of success happened in perception tasks, which are speech recognition, object detection, and machine translation. This wave also brought the revival of neural networks, but with more layers. Automation of pattern based on data e.g. credit approval, and the question could be answered like, should we extend credit or not.

Essence. The learning algorithm is constrained by the boundary based on the data. Some important questions are: where ML should be used? In a situation where we have uncertainty and we cannot pin the output down with mathematics. We should use ML when there is a pattern to be found, and in most situations, we do not need to know the pattern because the algorithm can find them. Also to apply ML it is essential to have the representative data.

Another important issue, NN has a very large number of parameters but it works well although it can be larger than the number of observations. NN algorithms generalize well even when we do not have a large data set. The still open question is: how is this possible that Neural Networks algorithm can perform so well and can be generalized to a great extent, even when the number of parameters is much larger than the available examples for training.

The revolution in machine learning. The 1st revolution in Machine Learning happened with increasing automation in NN. Humans are interfering less than before, but humans take part in NN's design architecture. The 2nd revolution came with the increase in computation speed. The use of GPU, in ML application made possible the processing 2 times as fast. Increasing the speed of computational processing created incentives for researchers to be more prone to experiments; experiment with different models, more than a situation where the process could take many days for example. The 3rd revolution came with data abundance. There were elaborate data resources, and the use of multiple data sources to profile a person. This time machines were being used to find patterns that could not be foreseen by humans, that is, find relationships that were not obvious. In the 4th revolution, it can be seen the rise of crowdsourcing, Tensor-Flow, GitHub, Google Colab, AWS. That is there are no barriers to entry in learning, and resources for computation are easily available. We can see that the main achievement of this newer revolution is superior intelligence in machines, in the sense of replicating humans and overcoming them. An eminent example is self-driving cars.

Challenges. Overfitting in Neural Networks. Practical challenges: bias in the algorithm introduced by humans, intentionally or unintentionally. And interpretability in models, this is a lesser issue because the capacity of a model to be interpretable is not needed sometimes.

There is the challenge of Security Risks, Machine Learning can be used for destructive or evil purposes.

And last but not least, Social Risks: machines can quickly replace humans in the job market.

Conclusion AI will replace routine intelligence (self-driving cars, e.g.) in the next 20 years. It is having a profound impact on the economy and security. AI is not Big Brother, there is no danger of revolution, not coming from machines at least. And we cannot ignore AI.

Session: Error Estimation for Pattern Recognition

Presenter: Ulisses Neto, professor at Texas A&M University, author of *Error Estimation for Pattern Recognition* (Braga-Neto and Dougherty, 2015)

It is important to distinguish deterministic models versus stochastic, and mechanistic versus empirical. One eminent example of a deterministic and empirical model is from Kepler's laws of planetary motion. This is an example because they are simple formulas that describe the motion of planets using observational records. On the other hand, Newton derived the same laws but using mechanistic reasoning based on physical principles. The latter is an example of a deterministic and mechanical model. And the first stochastic model is from the Least Squares method created by Gauss to analyze asteroids trajectories.

Deterministic and mechanistic models are not effective in very complex domains due to the presence of significant unexplained variability. Complex systems have unobserved values, which is a latent factor that affect the value target, the stochastic model are also appropriate to measurement noise.

The problem that Ulisses is interested in presenting is how to calculate the error rate of a classification model.

It is presented the concept of *apparent error* or *training error*, it is biased but Ulisses argues that this can be a good estimator in some situations. In most cases, there are few data and it is being applied a non-linear model. This may cause overfitting when done using calculating the apparent error. In this case, the method of calculating the error rate is not good because is biased.

According to Karl Popper in his book *The Logic of Scientific Discovery* (Popper, 2002), Science is base on testing predictions. In the case of interest here, the question is if the error estimator will predict correctly the error rate of the pattern model in all future samples.

It is important to note that a non-consistent estimator can also be useful, usually in situations where we have a small sample. As the scissor diagram shows an estimator that is not consistent can be more useful compared to a consistent estimator that has a bigger bias for small samples.

Concepts: *Error Estimation Rule* is a mapping $\Xi : (\Psi, S, \xi) \to \hat{\varepsilon}$; Ψ is a classification rule; S is a sample; ξ is random component; and $\hat{\varepsilon}$ is the error estimator.

We call a *pattern recognition rule* the tuple (Ψ, Ξ). And a realization of the rule given the data is a *pattern recognition model* (ψ, ε).

It is known that $\hat{\varepsilon}$ is a random variable and its conditional variance is $V = \mathrm{Var}[\hat{\varepsilon}|S]$. Using the conditional variance formula one gets $\mathrm{Var}[\hat{\varepsilon}] = \mathrm{E}[V] + \mathrm{Var}[\mathrm{E}[\hat{\varepsilon}|S]]$. It is defined as the *test-set error estimator* this is simply the error estimator estimated using a sub-sample of the data, which is the training sample; and the value is of the estimator (estimate) is calculated using the other complementary sub-sample which is called the test sample. This estimator has good properties, like unbiasedness, know Binomial distribution, and consistency. But the drawback is that if only a small sample is available, there will be insufficient data for the training and test samples.

To evaluate the Error Performance Estimator it is important to consider the classification rule, in which sample size the estimator is being used, the complexity of the model (dimensionality) and feature-label distribution (joint probability distribution). To compare the performance of different estimators we can measure bias, variance, and computational speed.

Most of the studies in this sense use simulations to measure those factors. An estimator for the prediction error should have the following properties: small bias (or be unbiased), small variance and below in computational demand.

The *Resubstitution* method of measuring the error performance simply counts and average the times that a classifier wrongly assigns an observation. As the formula suggests:

$$\hat{\varepsilon} = n^{-1} \sum_{i=1}^{n} |y_i - \psi(x_i)|$$

This error estimator is computationally simple, it is fast to calculate, hence it is preferred for large data sets. The estimator is non-random, which is given a sample it will give the same result every time. The problem with this estimator is that is usually estimated lower values for the error, that is, it is

optimistically biased. And this bias tends to be higher as the complexity of the model increases. But it is known that for a finite VC-dimension problem the Resubstitution method is strongly universally consistent, to prove this it used the Vapnik-Chervonenkis Theorem.

Cross-Validation method divides the sample into k sub-samples for a first-round one of the sub-samples is left out, the pattern recognition model is applied and the error estimator is used in the left out sub-sample, this procedure is applied for all k samples. The final error estimate is the average of all sub-samples. One can apply this method with different sub-samples many times, this can reduce the variance of the error estimator. Usually, this estimator shows higher values for the error rate than the true value, that is, the Cross-Validation estimator is pessimistic. The downside of the estimator is that it can have a large variance for small sample sizes. And it is computationally demanding.

In the *Bootstrapping* estimation method it is used the observed sample to make a resampling procedure, in which all observations have the same probability of being selected. This method makes the replacement of selected values, and the new bootstrap samples have the same size as the original sample. Note that there may be repeated observation due to resampling. In each bootstrap sample, the error estimator is calculated, then it is made an average of the error estimators of all bootstrap samples. In general, the bootstrapping estimator is negatively biased, has small variance and it is computationally intensive especially for large data sets.

The *Bolstering* is quite similar to Resubstitution but it is applied a kernel weight around the points. So that when a point is close to the decision boundary a part of the kernel is at the wrong side of the decision area, this situation penalizes points that are close to the decision boundary. The joint probability density function for the error estimator is:

$$f^{\diamond}(x, y) = n^{-1} \sum_{i=1}^{n} f_i^{\diamond}(x - x_i) \cdot \mathbb{I}[y = y_i]$$

where $f_i^{\diamond}$ is the kernel density chosen, and $\mathbb{I}[y = y_i]$ is the indicator function it assumes value 1 when the logical sentence inside brackets is true, and 0 otherwise. The Bolstering method has average speed, small variance, and small bias. To calculate the error estimator many times it is used Monte Carlo methods of integration.

Session: Big Data Systems and Analytics

Presenter: Ling Liu, professor at Georgia Institute of Technology, and IEEE fellow.

The presentation begins first with a discussion about the relationship between data science, mathematics and social impact. The sources of Big Data: On-line sources come from clicks at a site, billing events, server requests; also data generated by mobile applications usage, like social networks mobiles, Facebook, Instagram; another source comes from Health and Scientific computing.

The knowledge and insights that come from Big Data range from predicting and warning systems for earthquakes, car traffic management, or even disease-prevention; the advantages and good application of Big Data are evident. But there is also a dark side to it, given the giant amount of data we can support scientific hypotheses or theories that are not correct under a more judicious analysis.

In the age of Big Data, the important skills are to understand, to process data, extract value, visualize and also be able to communicate it to a wider audience. The main goal of Professor Liu is for the students of the school to be able to think critically about the data.

Big Data Systems. Big data can be characterized as data sets that use non-conventional methods for storage and analysis, including algorithms and hardware. However how people define and see what is Big Data is evolving and changing, this concept changes in different contexts of fields and periods, and what is common to be used in a certain area of data application.

The Big part of Big Data is formed by many challenges, first is the volume; when we include difficulties like non-structured data, dispersion into many servers and many agents accessing and transforming the data, the complexity of the task increases. In what concerns volume humanity is producing more and more data and in an accelerating rate, in 2009 all information of humanity was around 0.9 ZB (zettabytes) in just 10 years the amount estimated increased to 35 ZB. This increase in data volume came with or was a cause of the increasing instruments to collect data, nowadays all smartphones can collect huge amounts of information from the user, through e.g. location, browsing and purchase history. This increasing amount of data presents challenges for computation, communication, and algorithms to process and store this information.

It is not just the amount of information which is relevant and increasing but the speed with which the data is collected and processed. For example, sites that sell products for clients online need to calculate quickly incoming information to offer product promotions. Transport apps like Uber need huge amounts of storage and processing to execute their services.

Nowadays data is not just Big in volume but also in variety, there many and many more sources of data: video, audio, text, and multi-media. The different algorithms will recommend different decisions to take in a data-driven situation, in this sense, it is important to understand what are the pros and cons of each algorithm and to understand how do we compare those different recommendations. In this same line, we need to think about what are the different metrics to analyze a certain situation. Hence a data scientist needs to understand all those aspects of data analysis to make a better decision.

There is a bottleneck for the advancement of storage and analysis of data, they are primarily concentrated on the need for the creation of new architectures, algorithms, and techniques to store, process and analyze, also new challenges are appearing in security and privacy. Previously data sets were relatively small and structured, so it was easy to search for information. But now, information comes in huge amounts and an unstructured manner, so it is difficult to search for information in this kind of data set.

Graphs are structured data, they represent the connection of nodes that relate to other nodes, they can have directional and magnitude relationships. For Big Data applied in a graph structure they call it BigGraph. This kind of representation is used for road networks, internet, utility grids, and protein structure. Some social graph models have around 1 billion vertices and 100 billion edges, web network models have 50 billion nodes and 1 trillion edges, however, brain graphs have 100 billion nodes and 1 trillion edges. In the latter case, the most powerful machines available cannot process the network models from brains. The challenge now for computer scientists is to build an architecture to store BigGraphs in a light manner and with easy searching tools.

Some discrete optimization problems are mentioned. In a large data set, how to find the right information, finding, for example, the most important people in a network, monitoring algal blooms and reducing energy consumption in houses. Companies are interested in finding the most influential persons in a network, those are called hubs, because they are more effective targets for advertisement, they can more rapidly spread information through the market. Algae blooms monitoring is essential for sewage and utility companies, they want to know where are the best places in the system to install monitoring devices to detect water contamination. So the decision that must be taken here is to maximize a function constrained by a budget: $\max F(A) \text{s.t.} C(A) < B$ where A is the parameter to be changed and B is the budget constraint.

Robustness of Deep Learning Systems Against Deception

It is more common for companies to supply services where they can train your algorithm in a cloud platform, that is, firms can offer machine learning services in their servers. Where a client (which usually are firms too) asks for a server to which train their data. With this trained model and output from the algorithm, the client firms can attend their clients.

We can take an example where the client firm needs to train an algorithm for a self-driving car, where the algorithm needs to identify objects on the street. This can open problems related to attacks from hackers. Let us see an example with Adversarial Inputs, which are inputs given to an algorithm that will give the wrong output. This attacks algorithm can use e.g. gradient descent methods to find imperceptible changes for the human eye that can make an excellent algorithm point to the wrong answer. These malicious algorithms can have two objectives: they can make the target algorithm simply point to the wrong answer, or it can make the victim output a specific class.

Session: Deep Learning

Presenter: Atlas Wang, Assistant Professor at Texas A&M University (TAMU)

In neural networks, each layer extracts features from the output of the previous layer and all layers are trained jointly. The number of layers has been growing from 8 in 2008 with AlexNet, 19 layers in 2014 in VGG and ResNet in 2015 had 152 layers. The trend now is to build larger neural networks, deeper and with wider layers, train them with bigger data and using high-performance computing.

The main challenges in Neural Networks are the lack of theoretical understanding of the process and no theoretical guarantees of an algorithm's correct performance. There are some surprises in situations wherein principle intuition tells that the algorithm should not work. The training of the NN algorithm is computationally expansive and relies on many tricks and art/experience-based methods. And another

issue with the area is that there is are a rigorous and principled way to incorporate domain-based knowledge into the NN algorithm. In the same line, there is no principled way to interpret the behavior of the model.

Neural Networks are loosely inspired by the biological neurons, they have a structure in which electrical signal is sent between neurons given a certain threshold of each neuron, the complex interaction between this electrical signaling is how thought is formed by humans.

Epochs are the cycles that a NN passes in the training process. The training process in an NN consists of the machine receiving and input and then the output is tested against the true value or class. Then if the answer is wrong the algorithm weights are changed according to the error. The *Perceptron* is a linear classifier that is it has only the input layer and the output layer. For non-linear classification problems, we include more layers, called *hidden layers* those are between the input layer and the output layer.

For the training of the Neural Network, it is used the gradient descent. In each epoch, the weights are updated according to a loss function, and the gradient descent is the method for optimization. The *neuron* is the function that tells the activation signal of each node in the NN. The neuron needs to be (nearly) differentiable given that it is used gradient descent for optimization. An example of a differentiable neuron is the sigmoid, and a non-differentiable but continuous neuron is the ReLU.

From *fully connected* Neural Networks to *convolutional* Neural Networks (CNN). The convolution is a moving window kernel that can extract features in an input. There are many steps in a CNN but in each one, the algorithm will capture basic features to more specific ones as it goes to deeper layers.

Neural Networks algorithms use a mix of CNN and fully connected layers, for example, AlexNet in 2012 use 5 convolutional layers and 3 fully connected layers. Besides AlexNet uses ReLU, dropout and data augmentation.

The 3D Convolutional Neural Network (3D CNN) is used for taking into consideration the temporal aspect of inputs. So this could be used in videos, which is a sequence of images. In *visual attention mechanism* the algorithm can detect in the image the certain objects, so it can distinguish different objects in more complex images. The Graph Convolution Network (GCN) incorporates graph theory in the architecture of CNN.

In the Generative Models, the objective of the algorithm is to understand data through generations, this is a case of unsupervised learning. This is important because there are many unlabeled data than labeled data, so the machine needs to learn without any feedback, different categories of data.

Generative Adversarial Networks (GAN) this method uses two NN for a generation of models, for example, generation of images. One of the algorithms is the generator, this algorithm uses a random noise signal to generate images. The second NN is a discriminator, which will tell apart which images are good or close to real images and others that are not. This method has some problems: this algorithm is hard to train, it can be easily biased toward generative or discriminator side, and it hardly generates complicated images with many elements.

Activating functions are the instruction of how each node should be activated. There are many types of activating functions one of the most useful is the ReLU which is a linear function with a kink in zero, in ReLU the function has signal zero in negative values for input, then a linear 45-degree line function gives the output for inputs larger than zero. There are variants of ReLU, for example, Leaky ReLU and Randomized Leaky ReLU. Other activating functions include Sigmoid, Tanh, Softplus and, ELU.

In model parallelism, a Neural Network can be processed in multiple cores (multi-thread) of the processor, or many machines (message passing).

Some of the concerns in Deep Learning is the improvement in three aspects: i) Storage and Memory; ii) Speed (Latency), and iii) Energy Efficiency. The issue is that those three goals are not aligned, that is, for algorithm/hardware research it is important to find a good balance between the three, or to find an optimum for each kind of application.

Deep compression refers to methods that are used to simplify the algorithm, in situations where it can improve performance. One example of deep compression is pruning, where we randomly cut some synapses. This can make the training more efficient and have little cost in model accuracy.

Neural Architecture Search (NAS) is a promising field. NAS are algorithms that make better algorithms. A NAS, for example, can try to find optimum values for several filters, filter width, and filter height.

Session: Machine learning algorithms for making inferences on networks and answering questions in Biology and Medicine

Presenter: Alberto Paccanaro, full Professor at Royal Holloway University of London.

The question at hand is to find patterns in a large amount of unstructured data. Biological networks are natural representations of the interaction between biological and genetic phenomena.

A *graph* is a mathematical object used to represent pairwise relationships between objects. Each node or vertex is connected by edges or links. Graphs can be classified in: i) *directed graphs* and, ii) *undirected graphs*. In directed graphs, the links have a directional property that connects the nodes. In undirected graphs, links do not have the property of direction.

The Erdös-Rényi model is an algorithm for generating random graphs. Watts and Strogatz (1998) argues that many cases of important phenomena in the network sciences occur between complete randomness and complete regularity in graphs. The authors coined the term "small-world" networks which are specific cases of network architecture, which are between the two extremes of order and randomness. The power grid of the western United States and collaboration graph between actors are some notable examples of small-world networks.

A *degree of a node* is defined as the number of links that a vertex has. The *local cluster coefficient* (C_i) is defined at a vertex, and it is the ratio between the number of links between the neighbors of the vertex divided by the total possible combinations of links between the neighbors of the vertex. And the clustering coefficient of the entire graph (C) is defined as the average of all local cluster coefficients. The path *length* between two vertices is defined as the number of links in the shortest path between two vertices. The characteristic path length of a network is the average path length between all vertices (L).

The idea of Watts and Strogatz (1998) is a rewiring procedure. First, it starts with a regular network, then for each link, it is rewired with probability p. That is if $p = 0$ we are in a situation of complete regularity if $p = 1$ then we are in a situation of complete randomness. The question posed by the researchers is what happens with the measures of *path length* and *clustering coefficient* as p assumes values between 0 and 1.

With complete regularity, that it with $p = 0$, we find high levels of clustering (high C) and high values for path length (high L). When a network has a high characteristic path length we call it a *large world* network. On the other hand in complete randomness, $p = 1$, we observe a low clustering coefficient (low C) and low values of characteristic path length (low L). It is shown in Watts and Strogatz (1998) that it is not always the case where C and L are high in the same network. As p gets larger going from 0 to 1, C stays relatively stable in high levels while L drops abruptly. Only for p close to 1 that C starts to decrease. That is there is a large range for values of p in which C is high and L is low. Situations, in which there is a high value for clustering and low values for characteristic length, are called small-world networks. And the authors of the paper show that many real-world phenomena follow a small-world graph model.

In the Erdös-Rényi model, it is shown by Barabási and Albert (1999) that it is extremely rare to find vertices with high levels of connectivity, that is the probability of finding a vertex with connectivity level k decreases exponentially with k. In scale-free models there is a large probability of finding a vertex with high k, those are vertices that dominate the network. Scale-free networks are robust against random destruction or malfunction of one of the vertices. But it is extremely vulnerable against aimed attacks, that is, attacks that target specific vertices.

Network generation. In the real world networks generation happens with the addition of new vertices and links over time. And most networks exhibit preferential connectivity that is, there is a higher probability of new vertex to connect with a highly connected vertex. Those characteristics of real-world networks make them behave as small-world networks.

Basic biology. Proteins are made through the encoded information present in the DNA, which is translated into RNA then producing the specific protein. An assumption in bioinformatics is the *guilt by association principle* in which if two genes (or proteins) are activated at the same time as in a biological process, then we say that those two genes (or proteins) are connected.

In the network model for humans, we have about 25,000 nodes. Each node represents a protein (gene) and the edges represent the physical/chemical interaction between those proteins. This network is undirected. In a network model for the human biological process, we can see that there is a high degree of clustering. That is there are some notable important proteins in the network.

Usually, highly important proteins (hub proteins) are in an evolutionary sense slower change in time. Genes responsible for hub proteins are called essential genes, and those change and evolve slowly. In

protein interaction networks we observe small world phenomena because there are relatively short paths between any pair of nodes.

In the application of network sciences into medicine, we note that it is not easy to link a single gene to a disease, e.g. hundreds of genes can be linked to a type of cancer. And One gene can be linked to many diseases. When mapping diseases and genes in a network we observe that disease genes avoid protein hubs. And the same protein involved in disease tends to interact with each other. That is, we can identify a diseased neighborhood of proteins.

Session: Data Science in Astronomy

Presenter: Željko Ivezić, Project Scientist and the chair of the LSST Project Science Team.

Astronomy is about the search for life outside Earth and understanding the Universe.

The universe is expanding and this expansion is accelerating. A plausible explanation for this is dark energy and dark matter which is a source of gravitational force in the universe, another explanation is that the general theory of relativity is wrong.

One of the ways that astronomers study the acceleration of the expansion of the universe is through skymaps. The objective of the Large Synoptic Survey Telescope (LSST) is to make sky maps. Why are sky maps useful? Sky maps make a list of all detected objects in the sky, in this map, we gather information about size, color, the brightness of astronomical objects. Sky maps are important because they show new objects, for example, a new asteroid category, or star. Also, sky maps help in object classification, e.g. in the catalog of types of galaxies. Sky maps to aid in finding unusual objects, for example, a very weird star. And also it helps in measuring cosmological parameters, an important question is how fast the universe is expanding.

In making sky maps, astronomers make use of many types of instruments. Large telescopes can find faint objects. Telescopes above the atmosphere have high angular resolution (e.g., the Hubble Space Telescope) and other wavelength regions (X-ray, radio, infrared). And large sky surveys and sky maps make use of digital sensor technology (CCD: charge-coupled device), information technology (data processing and data distribution). And those telescopes make images of a great range of light wavelengths.

Why Big Data in astronomy? The Large Synoptic Survey Telescope (LSST) will detect 40 billion objects over half the sky. To make sky maps of stars and galaxies there is a need for large storage and process of large amounts of data. Given that sky, surveys can capture information for all the visible universe and in many points in time.

The objective of the Large Synoptic Survey Telescope (LSST) is to make a uniform sky survey. The whole observable sky will be scanned twice per night. And with 10 years of activity half of the sky will be imaged about 1000 times, that is, in the end, there will be a digital color movie in the sky. And in those 10 years, it is estimated that there will be approximately 100 PB of data. There will be measurements of 40 billion objects in the sky.

The main purposes of LSST lie on many endeavors in astronomy: in cosmology, astronomers are interested in identifying and studying dark matter and dark energy, the spatial distribution of galaxies and gravitational lensing. LSST will help us understand our galaxy's structure and also our solar system, in our solar system, for example, there are many unidentified asteroids and LSST will help us identify those objects. LSST will bring us data of about 40 billion stars inside our galaxy and 20 billion other galaxies. The challenges are Big Data processing and analysis.

It is expected that LSST will produce 20TB of data per day. LSST will capture pictures of the sky on many points in time. Hence it is possible to identify changes between images and identify fast-moving objects in space, for example, asteroids.

AstroML is a python module built by Željko collaborators. The motivation of AstroML comes from the fact that an essential feature of LSST and astronomy nowadays is the enormous quantity of data. AstroML will enable more people to analyze and study this data. Uniformity is another issue, with a single module it is easier to do more analysis.

Density Estimation, Clustering, and Classification One of the most important problems in data analysis is to discover $h(x)$ the density function of a certain random variable/vector X. The problem at hand is then: to find $h(x)$ or an approximation of it using only the data available, drawn from the random variable X. Statistics or estimators can help us identify or at least characterize the density function, some examples are: mean, variance, kurtosis, skewness, and percentiles. It is important to notice that those estimators are random variables, they are a function of the sample from X, and a combination of random

variables are per se a random variable. The estimators usually have Gaussian distributions when the sample is large.

Robust statistics In the case of a position statistics it is better to use the median instead of mean because the mean is an estimator too sensitive to outliers. In the case of dispersion statistic, the robust case is to use interquartile measures for example between the 75% and 25% percentiles, instead of using variance, which is also sensitive to outliers. The problem of using medians instead of means is that we have an increase in 25% of the uncertainty measure.

Gaussian Mixture. Let

$$h(x) = \sum_{j=1}^{M} \alpha_j \mathcal{N}(\mu_j, \sigma_j^2)$$

the random variable X follow a mixture of Gaussina distributions. Where α_j act as weights for the densities. The quantity of classes is determined by a penalized criterion for example BIC, or we can use cross-validation.

Supervised classification In supervised classification problems we know the outcome variable y with which we can train our model, and test it. And also we know before training the number of classes. There are two types of supervised classification models: *generative classification methods* , in which we make probability density estimation; and *discriminative classification methods* which makes decision boundaries between the classes.

Dimensionality reduction, Regression and Time Series In Principal Component Analysis (PCA) we try to find a linear combination of vectors in which they are orthogonal to each other. In PCA the eigenvectors of the matrix $X'X$, where X is the data matrix, are those linear combinations, the objective here is to model each X_i with a reduced number of principal components.

Non-negative Matrix Factorization (NMF) is used when there is no physical/theoretical reason for the variables to be negative. Hence this method impose positivity. About Independent Component Analysis (ICA): the only difference between ICA and PCA is that ICA requires independence between the components not just the absence of linear correlation. Locally Linear Embedding (LLE) is one of the most popular techniques for non-linear dimensionality reduction, this method uses local k-nearest statistical methods to model X. Compared to PCA, LLE needs considerably fewer components to make a satisfactory fit for the data.

Session: Data Science for Social Good

Presenter: Raymond T. Ng, associate director of the NSERC-funded strategic network on business intelligence. Professor of computer science at the University of British Columbia.

In his talk, Professor Raymond undertook three different themes: transportation, housing, and disease control.

Data science tools are used in all those three themes. Data science tools are used for integrating different sources of data, with special attention for unstructured data: text, images, geo-referenced data and so on. In-text data, for example, we can use data science to extract sentiment related data for modeling social phenomena. An important and previous part of modeling with data science and machine learning is the pre-processing and manipulation of large amounts of data, usually in a non-structured form. When analyzing private data the data scientist must be aware of ethical issues and individual privacy.

The boom of data science and machine learning came with the diverse and easily collectible data from cheap devices, for example, cameras, sensors, and geo-referenced data.

Transportation

Transportation data modeling is complex due to regional interaction, that is, it is important to consider data of many sources form different administrative regions and in many layers: social, demographics, economic data, etc. The metro Vancouver system goes through many cities so that the City of Surrey has its transportation system managed with other surrounding cities. The complexity also comes from different sources of data, and the period of collection of data; and the objectives of the project. The objectives in this project were: public transportation analysis, private vehicle greenhouse gas (GHG) emissions and electric vehicle adoption. For the public transportation analysis

Analysis of Riders' Tweets (2017) Three University of British Columbia (UBC) students: Saeid Allahdadian, Lap-Tak Chu, Mina Park, and William Qi analyzed the transit network and I studied patterns in social media data in Metro Vancouver.

In another study, researchers tackled the bus transportation system in the City of Surrey. One of the important aspects of studying transportation phenomena is the difficulty to make large scale surveys due to costs and time. So those are some examples of using data mining and web scrapping to gather data to model transportation phenomena. The study uses online posts from bus riders so that data is gathered instantly. With this tweet data we know the distribution of riders in different regions of the city. Through the day we know what is the flux of riders in the city and also what is the experience of the riders using the transportation system. They also link this data with different sources, for example, Translink trip planner and transit network.

In one of the projects presented in DSSG Andy Hong, Kevin Wong, Nimrah Anwar, and Mia Kramer study vehicle greenhouse gas emissions. They use vehicle stock data to study the distribution of GHG in the city of Surrey. They study how the types of vehicles (hybrid vehicles, electric vehicles) in the city affect GHG emissions.

Housing and Urban Development

Important challenges faced by the analysis of housing and urban development: housing ownership or rental, diverse sources of data, difficulty to measure the hidden rental market and the expansion of housing quality and quantity preserving biodiversity.

One of the projects presented in DSSG 2019 was about Preserving Biodiversity in Urban Development, the authors are Raghav Aggarwal, Lesley Miller, and Gabriel Smith.

According to Wikipedia: "Biodiversity encompasses all living species on Earth and their relationships to each other...". Some of the benefits of biodiversity are oxygen production, carbon sequestration, climate regulation, crop pollination and control of soil erosion. Some of the key threats to biodiversity in cities are housing and non-responsible urban expansion, hence habitat destruction. It is important before the expansion and planning of city building to understand and collect data about wildlife in your region if there are any species at risk, what are the types of ecosystems in your region and how species are distributed in space and time. To answer these questions we link datasets from many sources: climate, ecological, social, urban and biological information are merged to make a whole picture that is important for urban planning. Also in doing this analysis is important to consider flaws and gaps in the data. There can be biased in the representation of the data, some species are under-represented, or some areas of the map are under-represented.

Disease Control and Laboratory Testing In this section we explore some works done in DSSG in 2018 and 2019 about disease classification algorithms. Those works used semi-structured free form text data from lab reports containing raw test results. The objective of those works is to automate the prediction of diseases based on symptoms because the manual classification process is expensive and slow. The studies used natural language processing techniques for interpreting and labeling unstructured laboratory reports.

Professor Raymond show some examples in which we can use **Natural Language Processing** for sentiment analysis (in riders in Metro Vancouver) and term and topic extraction (in housing and disease classification). Now we see how NLP can be used for summarization.

The research in NLP until the past decade focused on formal documents, e.g. newspapers, reports, etc. But now it comes more to the attention of researchers' informal documents and evaluative texts, for example, emails, blogs, twitter, and user reviews. In this sense one of the objectives of NLP is to evaluate the sentiment of the writer/user: for example, examining a product review identify if the client was satisfied or not with the product and what are the issues and positive aspects of the product. Some of the key differences between formal and informal texts are that informal texts are usually very short, with looser grammar, they can contain more misspelling and can be in great quantity.

In the product review example, the NLP researcher is interested in identifying which are the terms and words that are associated with positive and negative aspects of the product, and which parts or characteristics of the product are good or bad. This is the extraction of features or feature mapping, the taxonomy of the product's characteristics are important so that it provides to the user a conceptual organization of features and eliminates redundancy for the analysis.

Session: Scheduling in Warehouse-scale data centers

Presenter: Francisco Brasileiro, Full Professor at the Universidade Federal de Campina Grande.

State-of-Practice. Google, Amazon, and Microsoft are examples of large cloud computing providers. They operate warehouse-scale data centers where they offer storage, processing, and many more cloud

computing services. Each data center has a large number of clusters, and usually scheduling is made at the cluster level.

Some of the characteristics of warehouse-scale data centers are: There is a high degree of heterogeneity in clusters, that is, there are large differences between clusters in terms of CPU, memory capacity and other machine attributes. In a common cluster of Google's data centers, there are 67 unique machine attributes, some of the attributes are: chipset, SSD storage availability, number of disks, public IP, OS kernel version, etc. Those attributes are used to define constraints and capacities for a given *task*.

Google clusters receive hundreds of *requests* in a month. Each request is a set of *tasks*. Each task is defined by: i. service class; ii. resource demand; iii. duration; iv. constraints. The 12 different types of service classes are aggregated in 3 higher-level classes: i. "Prod" is the most important type of task, those are critical long-running services, for example, streaming, monitoring, and user-facing services; ii. "Batch" is an intermediate level of importance service, those tasks need a minimal level of Quality of Service (QoS); iii. "Free" is the lowest level of importance service, those are tasks associated with the testing phase of projects, and run only when there are available resources.

The heterogeneity in tasks has the following characteristics: Prod tasks take more CPU and time to process, Batch comes in second and Free is the least demanding. Usually, the submitted tasks do not have constraints, only 6% of the tasks have constraints and the majority of them only use 1 or 2 attributes for the constraint.

A *server* is defined as: $s =< id, rc, attr >$ a tuple. Where id is server identification, $rc = (c_1, ..., c_n)$ is a list with the server's attribute capacities and $attr$ are the attributes associated with the server. A *request* is also a tuple, in which we need to specify the service class (e.g.: Prod, Batch, Free), the set of *tasks* that comprise the request and the constraints associated with those tasks. In each *task* we specify software and hardware requirements to execute the task.

Some of the problems faced by cloud computing service providers are fragmentation of resources and resource stranding. The Service Level Agreements (SLA) of Azure, AWS and Google state rules about the minimum performance and eligibility of credit and discount in the service payment. Usually, the Service-Level Objective (SLO) of those service providers are based on time of connectivity.

The objective of professor Francisco Brasileiro and his team is to build a scheduling architecture that minimizes the problems of wasted resources and fairness among the same level tasks. In this scheduling, tasks that are surpassing the SLO will have some of their resources allocated to tasks that are not meeting the SLO. Their metric of QoS is *availability* defined:

$$A_t = \frac{a_t}{a_t + p_t}$$

where a_t is is the amount of time that task t has had resources allocated to it since its admission. And p_t is the amount of time that t has been in the pending queue since its admission. Tasks (in the same level) with lower A_t have priority. The task with shorter time to reach SLO and that have the same A_t will be allocated first. This is called the Time to Violate (TTV), this metric is only calculated for tasks with QoS greater or equal to SLO. For tasks with QoS smaller than the SLO we call the metric as *task recoverability*, tasks with QoS<SLO have priority in service. These metrics are calculated in the following way:

$$\frac{a_t}{\text{SLO}} - (a_t + p_t)$$

Professor Francisco Brasileiro and his team compare priority-based and QoS-driven schedulers, they use simulations for this. They show that the availability driven scheduler match especially for the Free type of task in many more cases than the traditional scheduler, that is the QoS of tasks are most of the time close to the SLO. In almost all circumstances the availability scheduler has less QoS deficit compared to the traditional priority-based scheduler. And in terms of fairness, the availability scheduler has a lower number of unfulfilled tasks (a lower level of inequality) than the priority scheduler.

Session: Data Engineering and Data Science

Presenter: Altigran Soares, professor at the Institute of Computing, Federal University of Amazonas (IComp/UFAM).

There is a huge interest in academia and industry in data science and data engineering. One issue faced by data scientists and data engineers is that around 80% of their time is spent on data preparation. That is the acquisition of data, extraction, deduplication, integration, cleaning, and protection. Of this

80%, 20% is used for the collection of data sets and the other 60% is used cleaning and organizing data. Data science and engineering is also an area with a huge workforce demand compared to supply. The time spent with data preparation is a great challenge for the data engineering research community.

So what is needed for organizations and enterprises that are faced with data-driven questions? The proposition is the automation of data engineering, manual intervention should be limited to high-level feedback. This is important to improve the efficient use of the labor force and prevent human prone errors.

Data Civilizer is a tool proposed by DSAIL - Data Systems and AI laboratory in joined by Qatar Computing Research Institute (QCRI) where the researchers an architecture able to automate many tasks related to data preparation. Some of the tools of Data Civilizer are data discovery, data stitching, cleaning, transformation and more (http://dsail.csail.mit.edu/index.php/data-civilizer/).

Session: Open Science and Research Data Management

Presenter: Cláudia Bauzer Medeiros, full professor at Unicamp.

Open science comprises access to papers, data and access to processes. In this sense, open science means all artifacts used in the scientific process are available in the public domain. Open data does not mean necessarily sharing everything is available. Open data means necessarily that anyone can discover the existence of the data and how to obtain it. All of this process under confidentiality, security, and intellectual property issues. Open science is important due to validation and reproducible research. Open access science can save resources because we have access to reusable data and processes. With access to this information, we can improve, modify and accelerate scientific research and technological innovation.

FAIR data means Findable, Accessible, Interoperable, Reusable. This is an important aspect of open science. FAPESP, state of São Paulo Foundation for Scientific Research, is working on open science projects with all major universities of the state. Some of the challenges that arise in data management planning are: which data will be stored, which formats and for how long. All of those constraints plus the ethical and privacy aspects. Why is it important to pursue open science? Accelerate science discovery, promote worldwide collaboration, open financial opportunities, and promote knowledge.

Session: Data Science Methods in Economics

Presenter: Sergio Firpo, full professor at Insper.

In the *decomposition methods* literature we are interested for example: In what are the determinants of wage differences between men and women? That is, what are the determinants of the gender wage gap. Our objective is to measure how much of the wage gap is due to explained variables such as difference is human capital, labor market experience and types of occupation. The unexplained part could be due to gender labor market discrimination or could be due to unobservable skill differences. To estimate the wage gap we can use OLS. This model can show us how much a woman would earn if she had the same degree of human capital, type of occupation and labor market experience as a man. In line with the decomposition methods is the literature of treatment effect, which was explored in Sérgio's presentation.

The question of interest for the more recent literature on decomposition is not only about differences in means but also estimating differences in distribution. For example, we note that in developed countries in the last 30 years an increase in wage inequality. The first question is what are the determinants of this increase in inequality, it could be for example an increase in returns to education. But by asking the change in distributions, we want to know e.g. if inequality increase for the top 10% of the income distribution, or increases more for the bottom 10%.

In decomposing effects in distributions, the analogy to quantile regressions does not work. We first need to start to think about counterfactual distributions. There are two main ways to estimate counterfactual distribution. One is by estimating quantile regressions for all quantiles and then invert. The second is by using a weighting factor to estimate the conditional distribution. To calculate the weights we need to estimate a logit of an individual being treated dependent on X, the covariates.

Now we consider the treatment effect methods: Let $Y_i(0)$ be outcome for individual i when it did not receive treatment. Let $Y_i(1)$ be outcome for individual i when it receive treatment. For example a worker being unionized or not unionized. We are interested in finding an unbiased estimate of $\beta_i = Y_i(1) - Y_i(0)$. That is, what are the effects of the worker being unionized or not in e.g. income.

For the same individual i we cannot observe simultaneously $Y_i(0)$ and $Y_i(1)$, either the individual was treated or not treated. In this sense we only observe $Y_i = T_i Y_i(1) + (1 - T_i)Y_i(0)$. Hence we know that $Y_i = Y_i(0) + (Y_i(1) - Y_i(0))T_i$.

If we could observe both outcomes for each individual we would be able to observe the individual treatment effect (ATE). But we do not observe both outcomes for each individual, we only observe the set: $\{Y_i, T_i\}_{i=1}^N$ where N is the number of individuals and X is the matrix of covariates.

We define the average treatment effect on the treated (ATT) as $\mathrm{E}[Y_i(1) - Y_i(0)|T_i = 1]$ this is the effect of treatment only on the individuals that were treated. This is relevant because it can be the case that the group of treated and non-treated are different. In the case of random experiments, there is no difference between the treatment group and non-treatment group (before the treatment), hence the average treatment effect is the same as the average treatment effect on the treated, ATE=ATT.

Under some circumstances, we can assume that we observe all variables that influence the status of the treatment of an individual. In this case, we can estimate the probability function base on X of the individual being treated or not. That is: $\mathrm{E}[T_i|X_i] = P(X_i)$. This probability is called the propensity score. When we can use only X to explain the treatment status T we say that we have *treatment ignorability* or also called *unconfoundedness*. If we can assume treatment ignorability then we can calculate the average treatment effect and average treatment effect on the treated.

We can estimate ATT using imputing, another way is using matching. In the matching approach first, we need to find for each treated individual a corresponding non-treated individual. For this, we choose the closest non-treated individual. The criterion of distance is the euclidean distance in the covariates vector X. We can also use a criterion of the mean of the k closest individuals that are not treated, and use e.g. the propensity score as the measure of distance. This can be done only if the sport (the domain of X) is common for treated individuals and non-treated.

In the case where we cannot assume treatment ignorability, we can use instrumental variables. For the instrument to be valid it must be independent of the outcome and treatment variables. And the instrument must affect the probability of an individual being treated only in one direction, that is, monotonicity. With this, we can compute a local treatment average effect (LATE). Local in the sense that is only valid for the compilers, which are the individuals that are affected by the instrument, and exclude individuals that are always takers and never takers, which are individuals that do change behavior due to the instrument. To use instrumental variables we assume that the treatment status is discontinuous in the instrument variable. And that the outcome variable (conditional on being treated or non-treated) is continuous in the IV. Those two assumptions are equivalent to saying that the instrument does not affect the outcome directly, but only through the endogenous variable.

One alternative in the case where we cannot assume treatment ignorability and we deal with aggregate data is using the method called *synthetic control*. An example of a synthetic control application is in a case where we have a region that is affected by a policy. We build a weighted average of similar regions (that were not affected by the policy) to find a counterfactual region to compare with the affected region. The advantage of these methods is that we do not need to observe all relevant factors and we can use any amount of regions to build the counterfactual region. Standard errors can be computed through bootstrapping.

Session: Machine Learning Life-cycle support

Presenter: Fabio Porto, a senior researcher at the National Laboratory of Scientific Computing (LNCC).

Data EXtreme Lab (DEXL) is a group of researchers at the National Laboratory of Scientific Computing - Petrópolis (RJ). The purpose of the group is to research and develop cutting edge technologies in machine learning, AI, Data Science, and Big Data.

It is important to manage the wealth of data and models while working in a machine learning product. One of the top concerns in ML application is to choose the right set of features and appropriately tune the model.

That is why exist machine learning life cycle managers: *ModelDB* is a management system for building machine learning life-cycle. ModelDB keeps track of different models run by the ML engineer, it supports models from Python's Scikit learn and Scala- Spark MLIB, and helps in model search, and visualization of models results. *ML flow* is another system for managing ML life-cycle. It helps the engineer to keep track of experiments and helps in the deployment phase. ML flow keeps track of model parameters, input data set, performance metrics and model code version.

In building an ML product since data collection to deployment, the machine learning training part is just a small fraction of the total workflow. Between the most time-consuming tasks are data collection, feature extraction, serving infrastructure and monitoring tools. To manage and keep track of all those steps in the making of an ML model it is important to use an ML life-cycle management system.

Session: How "Big Data" impacted Atmospheric Sciences?

Presenter: Pedro Dias, Professor of Institute of Astronomy, Geophysics and Atmospheric Sciences (IAG) of the University of São Paulo.

There were already many papers that applied machine learning methods to analyze patterns in climate phenomena. But there was limited success in predicting climate on a time scale beyond a few days. Ensemble methods and genetic programming are rich techniques that show improvements in forecasting.

In climate science, models face a huge problem which is non-linearity and number of parameters of the $10^{10\sim11}$ order of magnitude. For example, a model that describes the relationship deforestation and precipitation. Going beyond linearity we find that there is an increase in precipitation for the few percentage points increase in deforestation. But total pasture causes no precipitation. This is an example of a non-linearity relationship.

Climate is a complex system. We can think of some factors that impact climate: biochemical interactions, ocean-atmospheric-land interactions, greenhouse gas concentration, cryosphere interaction.

Recently there has been a boom in the quantity and quality of data in climate sciences. Some examples of data sources are Ocean data buoys, weather ships, surface stations, weather radar, satellite images and so on. It is projected that from all those sources the climate data will grow to 350PB by the year of 2030.

Now we can also think about crowd-sourcing data from the internet of things sources, where for example smartphones can capture atmospheric pressure and record climate events.

Session: Computing in the Continuum

Presenter: Manish Parashar, Office Director of the Office of Advanced Cyberinfrastructure (OAC) at the National Science Foundation (NSF), Distinguished Professor of Computer Science at Rutgers.

Society and technology are being profoundly transformed by data and processing capacity. Nearly all scientific discoveries are transitioning from a data poor to a data rich environment. Some examples of the impact of data and computing in science and technology are the LSST (Large Synoptic Survey Telescope) in astronomy, personalized medicine, the Internet of Things, DNA sequencing and many more.

In many applications, the enormous amount of data being produced by the Internet of Things makes sending data to cloud unfeasible. Cloud computing traditionally has been dealing with: storage, compute and analytics to extract value. In *distributed computing*, computation, storage, and analytics are performed in the cloud, in the edge and things (e.g. IoT devices). The advantage of distributed computing is that the device has the computational and storage capacity to deliver value, and not be dependent on cloud computing.

An example of where this is important is in self-driving cars. The latency requirements for a self-driving car to make a decision is limited. Hence it is not feasible to send data to the cloud and process than return to the car. There are high costs in using only processors at the core of the network. So we need to leverage resources from the edge. An important research topic is to use neural networks to approximate outputs e.g. loops and parameter interval reduction.

An example of an application for edge computing is in the early detection system for tsunamis. Seismometers need to identify the earthquake's features: magnitude, location, and speed of displacement. But seismometers are good to identify small scale earthquakes and high-precision GPS are used to identify larger earthquakes. Computing in the continuum can help researchers to combine those data sets to improve speed prediction and warnings.

References

Barabási, A.-L. and R. Albert (1999). Emergence of scaling in random networks. *SCIENCE 286*.

Braga-Neto, U. M. and E. R. Dougherty (2015). *Error Estimation for Pattern Recognition*. IEEE Press.

Krizhevsky, A., I. Sutskever, and G. E. Hinton (2012). Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 1097–1105.

Popper, K. (2002). *The Logic of Scientific Discovery.* Routledge Classics, first published in 1935 by Verlag von Julius Springer.

Watts, D. J. and S. H. Strogatz (1998). Collective dynamics of 'small-world' networks. *NATURE 393.*