Today's computer scientists are developing solutions around artificial intelligence. Deep learning, a subset of machine learning, which is also a subset of artificial intelligence, is a technology has revolutionized the way we make transactions, communicate, and come up with solutions that help in different industries. At its very core, Artificial Neural Networks, is one of the most popular computing technologies of today. Named upon resembling a human brain, neural networks rely on training data to learn and improve over time, simulating how humans think.

Upon exploring IEEE, there are few of the most significant related papers that has shaped some of today's technologies:

Name of Journal	**Publisher**	**Years in Publication**	**Description/Abstract**
Deep Residual Learning for Image Recognition	He et al., IEEE Conference on Computer Vision and Pattern Recognition (CVPR)	8 years (2016)	A residual learning framework is proposed to ease network training. The model is easier to optimize and is more accurate in a considerably increased depth. Due to its extremely deep representations, the model is able to compete against other models, with a 28% improvement using the COCO object detection dataset.
Densely Connected Convolutional Networks	Huang et al., IEEE Conference on Computer Vision and Pattern Recognition (CVPR)	7 years (2015)	DenseNet connects each layer in a feed-forward fashion that addresses the vanishing-gradient issues, create a more concrete feature propagation, encourage feature reuse, and reduce the number of parameters.
Going Deeper with Convolutions	Szegedy et al., IEEE Conference on Computer Vision and Pattern	9 years (2015)	A deep convolutional neural network, codenamed Inception, achieves the new state-of-the-art classification and detection technique for

	Recognition (CVPR)		the ImageNet Large-Scale Visual Recognition Challenge 2014 (ILSVRC14). The proposed model has an improved utilization of computational resources.
Long Short-Term Memory	Hochreiter & Schmidhuber, Long Short-Term Memory. Neural Computation	27 years (1997)	A gradient-based method that bridges minimal time lags by enforcing constant error flows through constant error carousels within special units. LSTM runs with a computational complexity per time step and weight of O(1). LSTM produces more successful runs compared to other recurrent neural network techniques like Elman nets, back propagation through time, and neural sequence chunking.
Fully Convolutional Networks for Semantic Segmentation	Long et al., IEEE Conference on Computer Vision and Pattern Recognition (CVPR)	9 years (2015)	A fully convolutional model that is able to take input of arbitrary size and produce an correspondingly-sized output that is efficient in inference and learning. The paper defines the fully convolutional networks, explain their application to classification tasks, and draw connections to prior models. The model is adapted from AlexNet, VGGNet, and GoogleNet for transfer learning.

Upon researching those select few neural network-related articles, which by the way are very abundant from the IEEE website, it is very evident that the concept of neural network has existed long before this generation, as also seen in the Long-term Short-term study by Hochreiter & Schmidhuber (1997). Up to this day, LSTM remains to be used in developing Artificial Neural Network-based applications of today, more particularly in handwriting and speech recognition.

As described in the paper by Jain (1996), Artificial Neural Networks are defined as a huge parallel system with a large amount of interconnected processors. Artificial Neural Networks are designed to solve problems pertaining to pattern recognition, prediction, optimization, and associative memory. Artificial Neural Networks go beyond the norms of its time, providing a greater extension to conventional problem-solving approaches.

Artificial Neural Networks existed since the 20th century during a breakthrough research by McCulloch and Pitts. They laid the groundwork for the Artificial Neural Networks we know today. However, this technology remained within the shelves for 20 years, only to be revived in the 1980s where it received its renewed interest and development, as pioneered by Werbos' back-propagation learning algorithm for multilayer perceptrons and Hopfield's energy approach in 1982. Like its base term, neural networks, Artificial Neural Networks draw inspiration from the human brain, where neurons, as it is with Artificial Neural Networks, plays the role as building blocks of the brain. Artificial neurons, which can also be referred to as nodes, process and transmit information.

Artificial Neural Networks can be grouped in to two architectural categories: (1) feed-forward or loop-less, and (2) recurrent or looping due to feedback connections. Feed-forward networks are also referred to as multilayer perceptrons where neurons only produce a single set of output values rather than multiplicative ones. This is why it is also referred to as static or unidirectional. Recurrent networks, on the other hand, are dynamic wherein, due to its feedback connections, each input to each neuron is modified and entered to another state. Artificial Neural Networks have three learning paradigms: supervised, unsupervised, and hybrid. Supervised learning makes use of the correct answer or labels as input for learning. Unsupervised learning does not make use of the correct labels to learn the patterns. Hybrid combines them both.

References:

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep Residual Learning for Image Recognition. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). doi:10.1109/cvpr.2016.90

Hochreiter, S., & Schmidhuber, J. (1997). Long Short-Term Memory. Neural Computation, 9(8), 1735–1780. doi:10.1162/neco.1997.9.8.1735

Huang, G., Liu, Z., Maaten, L. van der, & Weinberger, K. Q. (2017). Densely Connected Convolutional Networks. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). doi:10.1109/cvpr.2017.243

Jain, A. K., Jianchang Mao, & Mohiuddin, K. M. (1996). Artificial neural networks: a tutorial. Computer, 29(3), 31–44. doi:10.1109/2.485891

Long, J., Shelhamer, E., & Darrell, T. (2015). Fully convolutional networks for semantic segmentation. 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). doi:10.1109/cvpr.2015.7298965

Szegedy, C., Wei Liu, Yangqing Jia, Sermanet, P., Reed, S., Anguelov, D., … Rabinovich, A. (2015). Going deeper with convolutions. 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). doi:10.1109/cvpr.2015.7298594

M2 Research 3: Genetic Algorithms

Genetic algorithms are a family of computational models. If a neural network is structured like a human brain, genetic algorithms encode solutions to problems through a chromosome-like structure and apply recombination operators to preserve critical information (Mathew, 2012). It is also referred to as an optimization tool for its ability to optimize search in a genetics selection principle. Genetic algorithms are all out for evolution, aiming to develop solutions that can be used for more and consecutive generations (Lambhora et al., 2019).

Upon exploring IEEE and other sources from the Internet, here are few of the most significant software applications of genetic algorithms:

Software	**Author/Publisher**	**Language**	**Functional Description**
Distributed Evolutionary Algorithms in Python (DEAP)	François-Michel De Rainville, Félix-Antoine Fortin, Marc-André Gardner, Marc Parizeau, Christian Gagné	Python	An evolutionary framework for rapid idea prototyping and testing, different from the usual black box testing. It incorporates parallelism, which deviates from traditional implementation methods such as synchronization and load balancing. DEAP makes use of a Python scripting language that glues the higher level Evolutionary Algorithm components into coherent Evolutionary Computation systems. DEAP has the following objectives: 1. Rapid prototyping without any compromise 2. Allow straightforward parallelization

			3. Preach by using a set of real-world examples
OptaPlanner	Red Hat, Apache License 2.0	Java	OptaPlanner is a lightweight, open-source, constraint satisfaction engine for optimizing planning problems such as vehicle routing, employee rostering, job scheduling, and task assignment. OptaPlanner supports high-level API for defining and supporting optimization problems and algorithms such as Genetic algorithms.
Genetic Algorithm Toolbox for MATLAB	MathWorks	MATLAB	This tool facilitates the implementation and execution of genetic algorithms in the MATLAB environment. This allows engineers, data scientists, and practitioners to visualize and demonstrate the different optimization techniques within the MATLAB environment.
Watchmaker	Daniel W. Dyer, Apache License 2.0	Java	A Java-based framework to demonstrate evolutionary computation. It supports a multi-threading engine and is non-invasive.

Among the software and tools discussed above, the Distributed Evolutionary Algorithms in Python (DEAP) is by far the most popular due to its versatile nature. DEAP is commonly used in experimenting with evolutionary algorithms. So, its main use are for optimization problems, parameter tuning, feature selection, dimensionality reduction, and automation. Its architecture is composed of three core components or modules: (1) base, (2) creator, and (3) tools.

The base module contains the objects and data structures of Evolutionary Computing that are not in the Python standard library. The creator stands as the meta-factory for creating classes through inheritance and composition in a functional programming paradigm, allowing attributes to be dynamically added when creating new object classes that a user specifically needs. Lastly, the tools contain the frequently used evolutionary algorithm operators which perform the statistics-related tasks.

DEAP can be applied in optimizing supply chain logistics. For example, for a company to operate in multiple warehouses and distribution centers, it must consider the most efficient and cost-effective route for transporting goods from one location to another. DEAP can be applied by the following process:

1. Define the representation scheme for candidate solutions that represent a distribution plan
2. Design a fitness function to evaluate the quality of each distribution plan
3. Implement the genetic operators required for this problem
4. Initialize a population of candidate solutions that represent different distribution plans
5. Applying DEAP's genetic operators and selecting the fittest individuals to produce offspring
6. Define a termination criteria for the GA
7. Integrate the DEAP into the company's existing software architecture
8. Deploy the optimized solution for execution and monitoring

The process defined above will enable the company to explore and optimize the ever-complex supply chain logistics while considering a variety of variables and constraints. Through the GA employed in DEAP, the company will be able to identify the most efficient distribution strategies to reduce costs and delivery times, and improve the overall customer satisfaction. In addition to that, DEAP is also very flexible. Meaning, it allows users to experiment with different optimization strategies and settings to locate the most effective solution to a problem such as this.