Deep Learning Applications for Modeling, Simulation, and Training

Tim Woodard
NVIDIA
Virginia Beach, Virginia
twoodard@nvidia.com

Mike Enloe
U.S. Army Combined Arms Center
Ft. Leavenworth, Kansas
michael.r.enloe.civ@mail.mil

ABSTRACT

With the explosive growth of Big Data and the emergence of general purpose computing with graphics processing units, Deep Learning has seen rapid adoption in many industries including healthcare, finance, entertainment, architecture, engineering, social media, speech recognition and translation, retail, advertising, high-performance computing, robotics, and transportation. Deep Learning is a rapidly growing class of Artificial Intelligence which has shown tremendous promise for solving heretofore intractable problems.

To date, the adoption of Deep Learning techniques has been somewhat limited in the Modeling, Simulation, and Training (MS&T) community. However, there are many possibilities for leveraging the work from other industries for the benefit of MS&T. Much of the active research in Deep Learning has direct relation to problems faced by the MS&T community, including image classification and segmentation, texture synthesis, physically-based material generation, ray tracing, style transfer, natural language processing, gaze tracking, character animation, and human behavior modeling.

In this paper, we will first provide an overview of Deep Learning concepts and how it fits into the broader contexts of Artificial Intelligence and Machine Learning. We then present the state-of-the-art in Deep Learning, including convolutional neural networks, generative adversarial networks, and phase-functioned neural networks. Finally, we discuss some of the ways that advancements in Deep Learning can be applied to meet current and future requirements in MS&T.

ABOUT THE AUTHORS

Mr. Tim Woodard is a Senior Solutions Architect with NVIDIA's Professional Visualization group. He has over 20 years of experience designing and developing software architectures for real-time visual simulation systems using modern GPU-based methods. Mr. Woodard has received patents for run-time simulator database generation technology and has published and presented papers at NVIDIA GPU Technology Conference, I/ITSEC, IMAGE, ASQ, and ITEC.

Mr. Mike Enloe is a Department of the Army Government Civilian Chief Engineer for the Synthetic Training Environment (STE) Cross Functional Team (CFT) at the Combined Arms Center in Fort Leavenworth, KS. He has worked at the NSC since 2001. His mission is to serve as the lead technical advisor to the Director of the STE. Current duties include designing concepts, engineering solutions, and directing S&T R&D to establish the next generation Synthetic Training Environment (STE). The STE converges Live, Virtual and Constructive simulation capabilities into a futuristic platform that enables Soldiers to maintain overmatch against our adversaries in the future. Past accomplishments include supporting development of the mission rehearsal simulation that planned Operation Iraqi Freedom, engineering middleware that links Army & Joint training simulations to real Mission Command Systems and the establishment of the U.S. Army Games for Training Program. Mr. Enloe has a Bachelor of Science in Information Technology degree from Phoenix University.

Deep Learning Applications for Modeling, Simulation, and Training

Tim Woodard
NVIDIA
Virginia Beach, Virginia
twoodard@nvidia.com

Mike Enloe
U.S. Army Combined Arms Center
Ft. Leavenworth, Kansas
michael.r.enloe.civ@mail.mil

BACKGROUND

Simply stated, Artificial Intelligence (AI) is the use of computers to simulate human intelligence. Apart from evoking ideas of futuristic sci-fi fantasies, AI has many real-world practical uses. Initially, AI was implemented by way of hand-coded algorithms based on classic flow-control mechanisms such as "if-then-else." For example, in the 1950's and 1960's, AI was utilized to play the games of checkers and chess. This was possible because of the well-defined rules and limited possible game configurations which bounded the complexity of mapping from the input (i.e., current game state) to the output (i.e., next move). As the amount of data and complexity of the problems increased, the methods utilized by computer scientists also increased in complexity. To address this complexity, a data-driven approach, as opposed to an exclusively rules-driven approach, was developed. Machine Learning and Deep Learning are categories of AI which help to overcome the limitations of human-defined rule-based approaches, as shown in Figure 1.

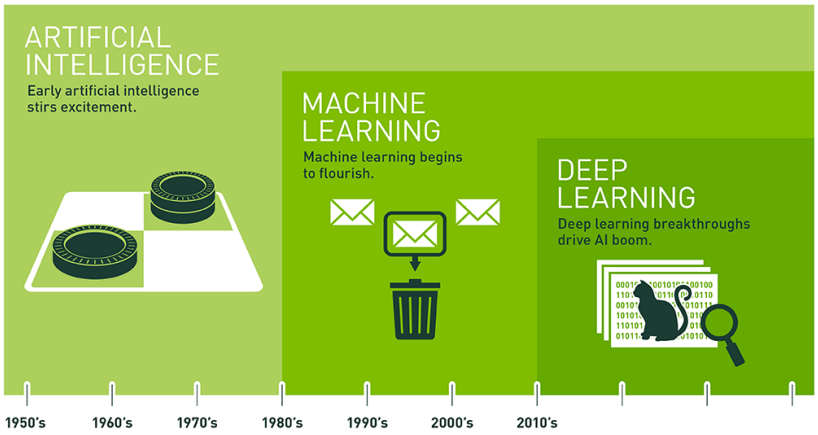

Figure 1 - AI vs. ML vs. DL

Machine Learning (ML) is a subset of AI that utilizes feature classifiers to find patterns in data to derive predictions. Common methods include clustering, linear regression, support vector machines, and Bayesian networks. A good example application of this technique is in the early implementations of email spam filters, which were based on bag-of-words models. The likelihood of certain word combinations was derived from data that represented legitimate emails vs. unwanted emails containing unlikely words like viagra, v1agra, vi@gr@, v-i-a-g-r-a, and so on. When applied to tasks like imagery classification and object detection, traditional ML-based methods require human-defined feature classifiers that the ML algorithms use for performing pattern matching.

Artificial Neural Networks represent a further subset of ML and are loosely based on the concept of biological neurons in that they "learn" based on repetitive training, eliminating the need for explicit rules. Neural networks are composed of an input layer for receiving data, one or more hidden layers composed of a set of basic operations and weighting values, and an output layer which indicates the resulting classification probabilities. As the number of hidden layers and weighting values increases, so can the potential efficacy of a network model.

Deep Learning (DL) is a subset of ML that seeks to leverage this fact by employing Deep Neural Networks (DNN), which contain many (dozens or even hundreds) of hidden layers – hence the name *deep*. Unlike other more general ML approaches, DL uses labeled data to automatically discover the distinguishing features which define a given classification *from the data itself*. By default, the weighting values in the network layers are randomized. The weights are computed in a process called *training*, which utilizes large amounts of labeled data and a method called backpropagation to reinforce or weaken the weights in response to correct or incorrect predictions. Once a network has been trained, it can then be used for *inference*, where it is presented with new data with which to make predictions.

Although neural networks are not a new concept, DNNs are relatively new. In 2012, Alex Krizhevsky used a deep convolutional neural network trained on GPUs to win the ImageNet competition by a significant margin (Krizhevsky 2012). Modern DNNs can contain millions of parameters and require a very large number of computations in the training process. Given these computational requirements, DNNs would not have been a practical approach until recently, with the availability of large datasets needed for training, and GPUs capable of accelerating the computations. So, while the Machine Learning field has been active for decades, Deep Learning has grown very rapidly over the last five years.

Preconditions for Deep Learning

Before exploring possible use-cases of Deep Learning for MS&T, let us first consider the necessary preconditions, namely data, algorithms, and massively parallel computing hardware. As we have seen, Deep Learning is fundamentally data-driven, and thus requires copious amounts of data to train a network effectively. In most cases, the data needs to be labeled to indicate to the DNN the expected mapping from input to label. Secondly, a deep learning software framework is necessary. There are many freely-available software frameworks that implement DNNs, including Google's TensorFlow, Facebook's Caffe2, Microsoft's Cognitive Toolkit, and many others. Finally, because the number of weights in modern DNNs can reach into the millions, and the training datasets are so large, the number of computations required for effectively training a DNN can be substantial. Therefore, the third precondition, parallel computing hardware, becomes necessary for training times to be sufficiently short enough to be practical. GPUs are an effective platform for general purpose parallel computing, as described in the following section.

General-purpose GPU Computing

The commonly known Moore's Law generally indicated that transistor counts in integrated circuits doubled every 18 months. For several decades, the result of this was that single-threaded applications could enjoy greatly improved performance with each new generation of a Central Processing Unit (CPU). Due to the physical limitations in the possible size and spacing of transistors, as well as the challenge of dissipating the heat produced, this trend tapered off. Year-over-year CPU performance no longer brings the 50% improvement it once did before 2005. These trends are described by Stanford's John Hennessy, who points out that the key to greater performance in the future is the use of "domain specific architecture" – processors which are purpose-built for solving specific classes of problems (Hennessy 2017).

In contrast to CPUs which may have dozens of highly complex cores, GPUs are composed of thousands of relatively simple cores designed for accelerating a specific set of mathematical operations. GPUs were originally designed for accelerating 3D graphics applications using a fixed-function pipeline but became programmable in the early 2000s via "shaders." This programmability was soon exploited for general-purpose computing tasks which could benefit from parallelization, and over the past decade remarkable performance gains have been achieved. As illustrated in Figure 2, over the next 7 years there will be a projected 1000x advantage over CPUs by 2025.

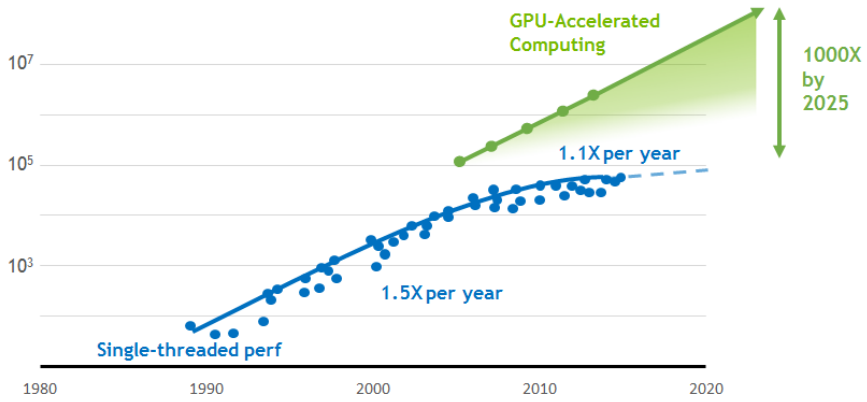

Figure 2 - CPU vs. GPU TFLOP Performance

It is important to note that legacy algorithms and software often cannot benefit from parallelization without significant modification. This is largely due to Amdahl's law, which reveals that even small amounts of serialization in an algorithm greatly limit how well performance will scale with additional processing cores. As Herb Sutter points out in (Sutter 2005), “the free lunch is over”, indicating that a very different computing model must be utilized to realize the potential improvements in performance in the context of parallelization, and it is not applicable to all problem domains. In this respect, GPU-based computing does not replace CPU-based computing for all problems, but rather accelerates those algorithms that can benefit from massive parallelization. While computer graphics is perhaps the most obvious benefactor of the GPU architecture, Deep Learning has emerged as another key application that requires massive parallelization to be practical.

APPLICATIONS OF DEEP LEARNING FOR MS&T

The primary reason that DL has proven so successful is that it provides solutions to problems which were heretofore intractable when applying traditional algorithms. The result is that in the past five years we have witnessed an explosion of growth with respect to the application of Deep Learning across many industries. These include healthcare, finance, entertainment, architecture, engineering, social media, speech recognition and translation, retail, advertising, high-performance computing, robotics, and transportation.

To date however, DL has not yet seen significant adoption within the MS&T community. A good strategy is to look for correlation between existing MS&T workflows and challenges, and DL-based techniques that are used in other industries to solve similar problems. Examples include content creation, realistic rendering, and behavior simulation.

Types of Deep Neural Networks

New use-cases for Deep Learning and new types of neural networks are emerging on an almost-monthly cadence, as evidenced by the number of academic papers published on sites like Cornell University’s arXiv (see https://arxiv.org/). A few of the more well-established network types are described below, along with general guidelines for where each applies.

Convolutional Neural Networks (CNNs) are well-suited to image-based applications. As the name implies, the primary operation in a CNN is convolution, where the values of neighboring pixels are used as input for computing a weighted sum that is passed on to subsequent network layers. Applications of CNNs include image classification, object detection, and semantic segmentation, which are particularly relevant to numerous MS&T content-creation tasks.

Recurrent Neural Networks (RNNs) are a class of neural networks for processing time-series data. In contrast to CNNs, RNNs are largely comprised of *long short term memory* (LSTM) and *gated recurrent units* (GRU) operations. Applications for RNNs include speech recognition, image captioning, and natural language processing and translation. These can be used for emulating human speech and motion.

Autoencoder neural networks utilize unsupervised learning and can reduce the dimensionality of a dataset. They can be used to automatically determine segmentation classifications and then perform a mapping from these classifications back to a representation in the form of the original input. Example uses of autoencoders are image denoising, super-resolution, anti-aliasing, and image colorization.

Finally, *Generative Adversarial Networks* (GANs) utilize two networks in parallel to automatically learn the distribution of the input dataset. For example, one network can be configured to produce realistic results, while a second network is configured to evaluate these results. This configuration allows the two networks to improve each other: the first network gets better at producing realistic output, while the second network gets better at distinguishing between real and generated data. This approach can be utilized in style transfer, which we describe in more detail below.

Content Creation

A simplified view of the typical MS&T content creation and visualization workflow, from geographic source data to visualization in a simulator, is depicted in Figure 3. Database generation is the process used to create a 3D representation of the world from source data. Image generation is the process of utilizing these databases for runtime visualization. In addition to image generation, other processes may also consume databases, including sensor simulations, and semi-automated force (SAF) systems. The following sections describe several ways in which Deep Learning can be applied to automate synthetic environment creation and increase visual fidelity.

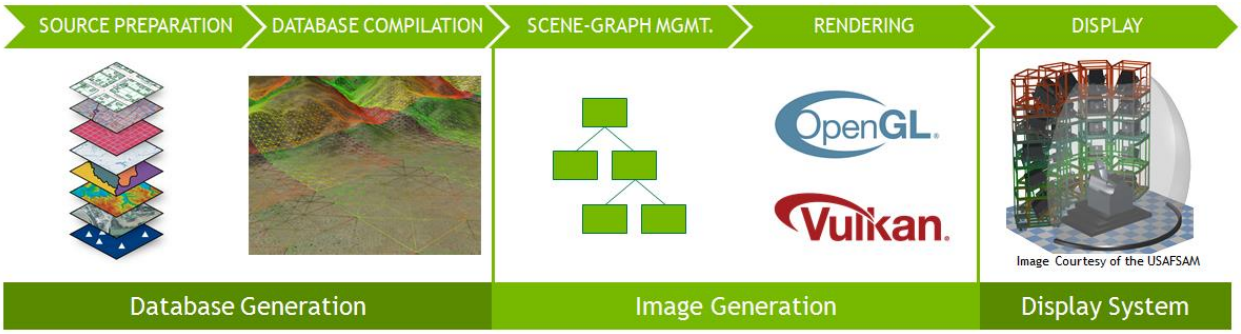

Figure 3 - High Level Visualization Workflow

Imagery Classification

One area where Deep Learning can be applied with respect to terrain generation is imagery classification - a problem domain for which Deep Learning is extremely well-suited. Imagery classification is concerned with determining an overall classification for a given image. For example, a DNN can be trained to recognize if a given tile of terrain imagery predominately falls into one of several categories, such as agriculture, residential, forest, urban, etc. When combined with other available data sources, this terrain imagery classification can be used as a basis for producing a rich semantic map of the world. As described below, this semantic map can be further utilized as an input for style transfer.

Imagery Segmentation

While image classification produces an overall category for a given image, segmentation seeks to classify individual regions or pixels within an image. Segmentation can be used as the basis for material encoding (required for sensor simulation), feature extraction (Yuan 2016), map generation, etc. For example, the SpaceNet dataset and GIS data have been used to train a network to perform building footprint and road segmentation (Barker 2016). The results can be vectorized and used as input for terrain database cultural feature generation. Note that the same method can be applied to multiple categories apart from buildings and roads, including vegetation, water, etc.

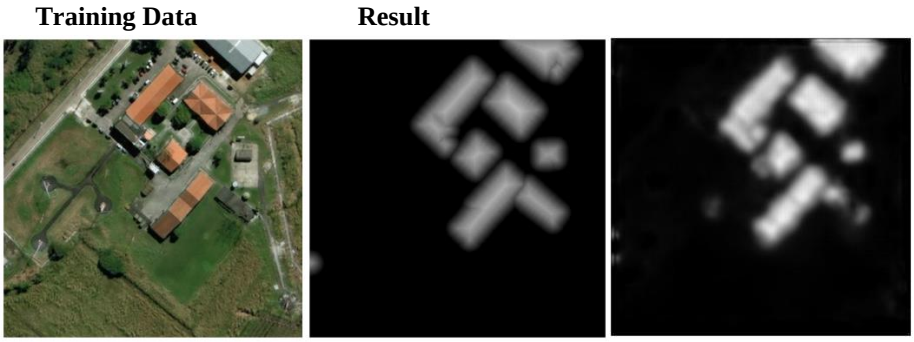

Figure 4 - Building Segmentation Training Data and Inference Result

Given a semantically tagged image, there are several applications that are highly relevant to this industry. For example, semantic images can be used as a basis for image synthesis based on a technique called style transfer. This is discussed in the following section. Another application of semantic tagging is sensor simulation, where not only information in the color-spectrum is important. A mapping from RGB pixels to material properties is possible using these techniques.

Imagery Synthesis

Another potential application of deep learning for improving the fidelity of virtual environments is style transfer. The goal of style transfer is to convert images from one style into corresponding images in a different style (Richter 2016). As an example, a DNN can be trained with images in two different styles: simple rasterized vectors (e.g., images from Google Maps), and aerial imagery. Once trained, the DNN can the take new images of rasterized vectors and perform style transfer, resulting in highly realistic and believable aerial images, as shown in the Figures 5 and 6.

Figure 5 - Example Training Data for Style Transfer

An application of this technique is the synthesizing realistic terrain imagery in the areas for which only low-resolution data is readily available. This has several significant advantages over traditional approaches. These include increased realism, reduced storage, improved semantic correlation, and congruity with adjacent high-resolution geo-specific areas.

Another application of DNN-based style transfer is for the generation of model textures. For example, semantic input that defines blocks of pixels in a handful of desired classifications (e.g., window, door, ledge) can be used to produce realistic building facades. Furthermore, the same semantic input images could be used to generate many different variations by applying different styles for each semantic category, as illustrated in Figure 7.

Figure 6 - Example Style Transfer for Synthesizing Aerial Imagery

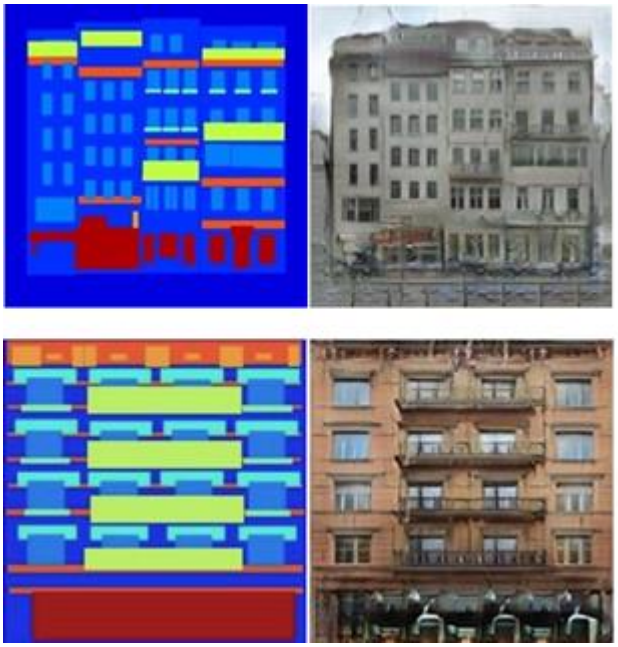

Figure 7 - Examples of Semantic Style Transfer for Generating Realistic Building Facades

This technique has even been successfully used to synthesize entire scenes, as show in Figure 8, where 2k-resolution photo-realistic images are generated from semantic labels (Wang 2017). This is accomplished by way of a generative adversarial network (GAN), which utilizes two competing DNNs configured to improve each other.

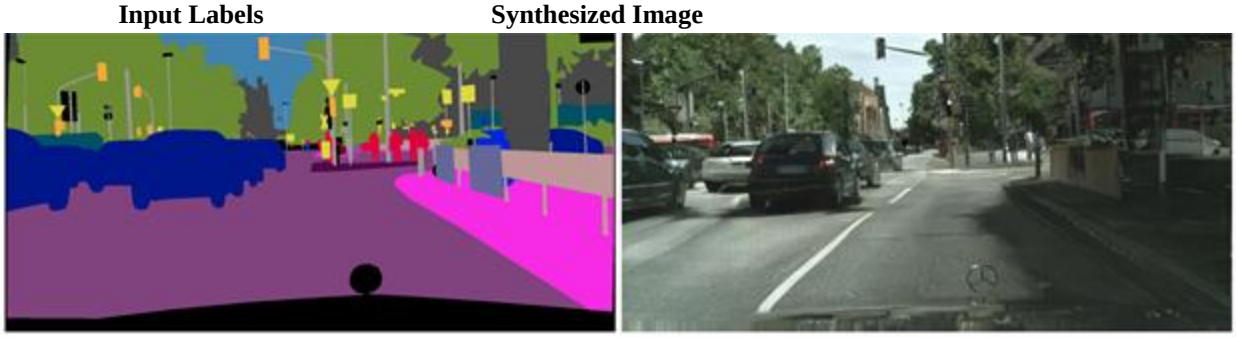

Figure 8 - Scene Synthesis Using Generative Adversarial Networks

Super-resolution

As display systems increase in refresh rate and resolution, it becomes necessary that the content used to produce virtual environments contains sufficient detail to avoid blurry or blocky images. As shown in Figure 9, DNNs have been applied to image super-resolution, also called upscaling, which is able to increase the resolution of images. This approach infers realistic details that are not present in the original image. Upscaling is currently used to "remaster" game art assets and has also been applied to full-motion video (FMV), taking as input 720p security camera footage and producing 4k video.

Figure 9 - DNN-based Upscaling

Realistic Rendering

Realism plays an important role in simulating synthetic environments as it helps to maximize immersion and reduces the likelihood of negative training. Physically-based rendering (PBR) is a method that utilizes material definitions that define how light interacts with surfaces, rather than relying on artists to produce these results manually or subjectively.

PBR is widely used for producing highly-realistic images in other fields that rely on ray tracing workflows, including architecture, engineering, product design, media, and entertainment, as shown in Figure 10. Defining PBR material definitions can be a daunting task, as it requires more information to accurately describe the properties of a given surface than with traditional material descriptions. Deep Learning is being used to automate the production of the material definitions needed for PBR from a single image (Aittala 2016), as Figure 11 shows.

Figure 10 - Physically-based Rendering using MDL

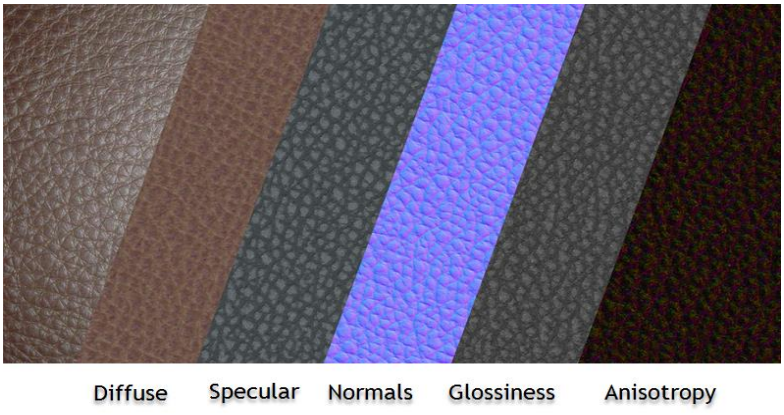

Figure 11 - PBR-ready Leather Material Generated from Camera Images

Behavior Simulation

Human Character Animation

Another area in which deep learning can be leveraged for the generation of realistic synthetic environments is human character animation. Traditional methods used to model synthetic entities often make it easy for the trainee to determine which entities are constructively generated. Ideally, the behaviors of live, virtual, and constructive participants would be indistinguishable. Three areas of research that can be used to achieve this goal are phase-functioned neural networks (PFNNs), natural language processing, and facial animation.

PFNNs can be utilized to generate natural-looking transitions between animation states, as shown in Figure 12. For example, as a character navigates complex terrain, it may need to climb, crawl, jump, side-step, etc., depending on the terrain and other environmental objects. In many traditional character animation systems, the transitions between these various states results in unnatural movements. PFNNs overcome this limitation, without requiring labor-intensive and time-consuming human motion capture (Holden 2017).

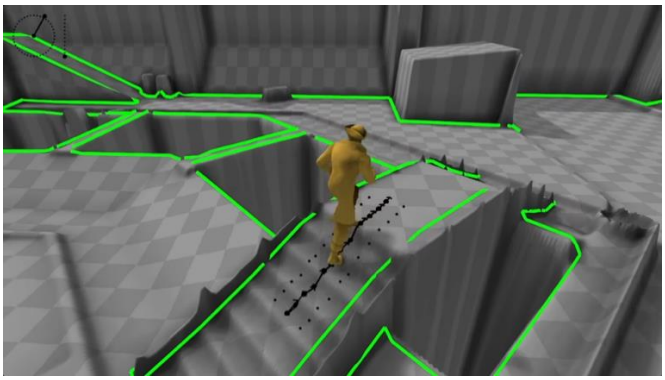

Figure 12 - Using a PFNN to Generate Natural Human Movements

Character Interaction and Natural Language Processing

Speech recognition, translation, and language understanding are all made possible with deep learning and is based on the use of RNNs, which are well-suited to problems in the temporal domain. Language is a good example of this, as the meaning of words is dependent on their sequence. Using RNN-based speech recognition, verbal commands given to constructive entities could be understood and carried out in the same way they would be by live participants.

As shown in Figure 13, A DNN has been trained to learn a mapping from input waveforms to the 3D vertex coordinates of a face model (Kerras 2017). This approach disambiguates the variations in facial expression that cannot be explained by the audio alone. During inference, the latent code can be used as an intuitive control for the emotional state of the face model. The network is trained with 3 to 5 minutes of high-quality animation data obtained using traditional, vision-based performance capture methods. The model yields reasonable results even when driven with audio from other speakers with different gender, accent, or language.

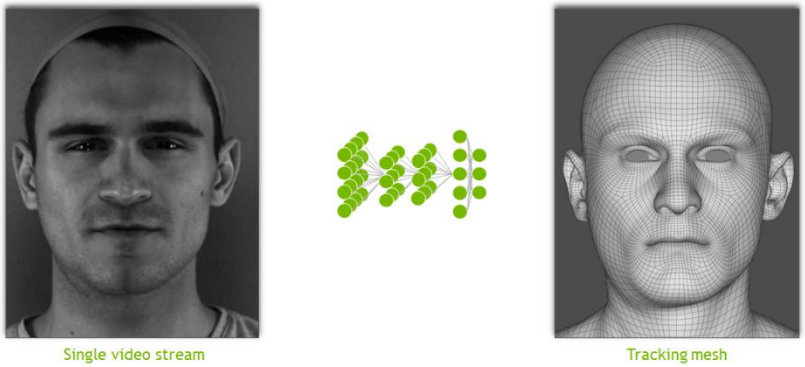

Figure 13 - 3D Facial Animation from Audio

DEEP LEARNING CHALLENGES

There are several challenges that can arise when attempting to apply Deep Learning. To be successful, it can be beneficial to start small, which provides iterative feedback sooner in the development process. Familiarity with the DL workflow, terminology, limitations, and applications can be overwhelming and takes time to acquire. Likewise, it can help to build on the work of others by starting with previously trained neural network models that align closely with the task(s) being performed. A key advantage to DL over traditional computer-vision algorithms is that a trained model can often be re-trained with new data and classifications in a relatively short amount of time. For example, a model that was designed to distinguish between images of dogs and cats can be quickly adjusted to classify other animal types.

Most often, data quantity or quality is a key factor since for deep learning to be effective, the data must be plentiful and well-curated. With respect to data quantity, it is sometimes possible to expand an existing dataset by performing transformations on the data, resulting in variations. For example, imagery data can be rotated, flipped, or supplemented with noise (Noever 2017). In cases where data labels are not available or are otherwise lacking, it can be possible to utilize other traditional ML methods to assist in defining data categories. Furthermore, there are DL-based methods that do not require labeled data, such as autoencoders described above. Finally, in some cases training imagery can be created through synthetic means, using simulation as a source (Cheng 2017). However, the efficacy of using simulated data for training neural networks can be highly dependent on the rendering fidelity. For this reason, physically based rendering approaches may be required.

CONCLUSION

We have given an overview of Deep Learning concepts and outlined numerous ways in which the advancements in Deep Learning in other industries can be leveraged for MS&T applications. Deep Learning is an exciting field which holds great promise in areas including content creation, rendering realism, and human behavior modeling. We believe that although Deep Learning is not currently mainstream within the MS&T industry, it has the potential to positively disrupt how realistic virtual environments are created.

REFERENCES

Aittala, M., Aila, T., and Lehtinen, J. (2016). Reflectance modeling by neural texture synthesis. ACM Trans. Graph. 35, 4, Article 65 (July 2016), 13 pages. Retrieved from: https://doi.org/10.1145/2897824.2925917

Barker, J., Gray, A. (2016). Exploring the SpaceNet Dataset Using DIGITS, NVIDIA Developer Blog, Retrieved from: https://devblogs.nvidia.com/parallelforall/exploring-spacenet-dataset-using-digits/

Cheng Z., MtCastle, T., Huster, T., Grattan, M., Camp, J., Cheng, H., Brisson, M. (2017). Human Activity Synthetic Data Generation. I/ITSEC 2017 Proceedings.

Hennessy, J. (2017). The End of Road for General Purpose Processors and the Future of Computing. 2017 Headlights Workshop on Advances in Computing Architecture, Stanford University, Retrieved from: https://systemx.stanford.edu/events/workshop/20170411/2017-headlights-workshop-advances-computing-architecture

Holden, D., Komura, T., Saito, J. (2017). Phase-functioned neural networks for character control. ACM Trans. Graph. 36, 4, Article 42 (July 2017), 13 pages. DOI: https://doi.org/10.1145/3072959.3073663

Karras, T, Aila, T., Laine, S., Herva, A., Lehtinen, J., (2017). Audio-driven facial animation by joint end-to-end learning of pose and emotion. ACM Trans. Graph. 36, 4, Article 94

Krizhevsky, A., Sutskever, I. and Hinton, G. E. (2012). ImageNet Classification with Deep Convolutional Neural Networks. NIPS 2012: Neural Information Processing Systems, Lake Tahoe, Nevada

Noever, D., Regian, J. (2017). Deep Learning for Training with Noise in Expert Systems. I/ITSEC 2017 Proceedings.

Richter, S. R., Vineet, V., Roth, S., & Koltun, V. (2016). Playing for data: Ground truth from computer games. In *European Conference on Computer Vision* (pp. 102-118). Springer International Publishing, (https://arxiv.org/abs/1608.02192)

Shanahan, J. (2017). Retrieved from: https://blogs.nvidia.com/blog/2017/11/01/gtc-dc-project-maven-jack-shanahan/

Sutter, H. (2005). The Free Lunch Is Over: A Fundamental Turn Toward Concurrency in Software. Dr. Dobb's Journal, vol 3, pp 30-33

Wang, T., Liu, M., Zhu, J., Tao, A., Kautz, J., & Catanzaro, B. (2017). High-Resolution Image Synthesis and Semantic Manipulation with Conditional GANs. CoRR, abs/1711.11585.

Yuan, J. (2016). Automatic Building Extraction in Aerial Scenes Using Convolutional Networks. CoRR, abs/1602.06564. Retrieved from: https://arxiv.org/abs/1602.06564