NVIDIA GPUs on Azure

An AI-First Infrastructure and Toolchain for Any Scale

October 2021
Table of Contents

Overview ....................................................................................................................................................... 3
The Motivating Need for an AI-First Approach .......................................................................................... 4
AI-First Infrastructure that Scales-Up-and-Out .......................................................................................... 8
AI-First Infrastructure at Any Scale ........................................................................................................... 11
Building and Operationalizing Models on AI-First Infrastructure .......................................................... 13
  Building Models .................................................................................................................................... 14
  Operationalizing Models ........................................................................................................................ 19
Summary ................................................................................................................................................... 23
Endnotes ..................................................................................................................................................... 24
Overview

AI has reached a critical juncture: the introduction of Transformer-class models for Natural Language Processing (NLP) are indicative of a quantum leap in model complexity. Driven of course by data, but now more than ever by model sophistication, this quantum leap in complexity has resulted in a corresponding requirement for IT infrastructure. Thus, the need for an AI-first infrastructure has been established – firmly established by NLP and increasingly so from other use cases (e.g., autonomous AI use cases such as self-driving vehicles). The only style of infrastructure capable of handling such demands is one that can not only scale up to take advantage of accelerators within a server, but also scale out to embrace many servers distributed across a network. Because computation and communication factor into training these next-generation models, Mellanox InfiniBand™ interconnected GPUs from NVIDIA have emerged as the de facto choice – a choice validated globally by a recent edition of the TOP500 list of supercomputers. Indeed, Microsoft Azure is the only public cloud that offers purpose-built AI supercomputers – i.e., massively scalable scale-up-and-out IT infrastructures comprised of Mellanox InfiniBand interconnected NVIDIA Ampere A100 Tensor Core GPUs. Whether it is to build and operationalize Transformer-based models for NLP or some other application domain, or to prototype a model that is much more modest, Azure is the right place to train and inference. Azure Machine Learning facilitates uptake of this AI-first infrastructure – from the earliest stages of development through to enterprise-grade production deployments that require full-blown MLOps. For any scale of AI workload, there exists a purpose-built AI first infrastructure on Azure – an AI-first infrastructure that optimally leverages isolated GPUs from NVIDIA to interconnected VMs fashioned into an AI cluster.
The Motivating Need for an AI-First Approach

From searches via Microsoft Bing to the productivity applications of Microsoft 365, the automated completion of words and even phrases is easily taken for granted. The technology behind predictive text is that of Natural Language Processing (NLP). With respect to NLP, of primary interest here are those advances made over just the past few years. In addition to prediction, these advances in NLP have enabled search engines like Bing to make the anything-but-subtle leap from keyword-driven queries to responses that demonstrate an understanding of the user’s intent.\(^1\) It is these recent advances that are fueling the utility of predictive text, question answering, document summarization, and more.\(^2\)

Written or spoken natural language is parsed sequentially as it is read or heard. This is as true for people as it is for traditional NLP – e.g., based upon popular approaches such as Long Short-Term Memory networks (aka. LSTMs).\(^3\) Of course, sequential parsing eliminates possibilities for leveraging parallelism. This debilitating limitation can be removed, however, with an attention-based mechanism for parsing natural language. Thus, the Transformer was introduced\(^4\) as:

“… the first transduction model relying entirely on self-attention to compute representations of its input and output without using sequence-aligned RNNs or convolution.”

In simple terms, Transformers enable the self-attentive parsing of natural language to be parallelized.\(^5\)
In 2017, the original Transformer model was trained on as many as 36 million sentence pairs for the purpose of language translation. Owing to parallelism inherent in its architecture, the authors of the original Transformer-based approach optimistically concluded: “For translation tasks, the Transformer can be trained significantly faster than architectures based on recurrent or convolutional layers.” Despite use of parallelism, however, the 300,000-step training runs for ‘big models’ took some three-and-a-half days to execute on eight NVIDIA P100 GPUs.

At roughly 17 billion parameters, Turing-NLG is second only to GPT-3 in terms of its complexity. Microsoft’s purpose in developing this Transformer-based approach was to advance ‘downstream tasks’ (e.g., question answering, conversational agents, and document understanding) via a generative language model. Importantly, Microsoft and NVIDIA have recently eclipsed GPT-3 with the 530-billion parameter Megatron-Turing Natural Language Generation (MT-NLP) model – a model that “… achieves ‘unmatched’ accuracy in a broad set of natural language tasks … including reading comprehension, commonsense reasoning, and natural language inferences.”

What has transpired is nothing short of remarkable: over the past four years, NLP has experienced generations of Transformer-based models. Simultaneous with this uptake of the attention-based paradigm for leading-edge NLP, has been an exponential growth in model complexity. That this growth has been literally exponential is made clear through the semi-log plot of Figure 1. Capturing data over just the past four years or so, model complexity has increased by three orders of magnitude or a factor of roughly 1,000. To underscore this point: ‘latest challenger’ MT-NLP is more than 5,000 times more complex than ELMo.

Given that Transformer-based approaches are really a recent introduction, and that they are already responsible for impressive advances in NLP, a prudent conclusion is that the trend evident in Figure 1 is likely to persist into the foreseeable future. Already the case from their
introduction in 2017, compute-intensive training of transformer-based models exceeds the capacity and capability of most IT infrastructures for Deep Learning in practical terms – even ones crafted with multiple NVIDIA Ampere A100 GPUs in a single-server configuration.

Moreover, complex models retain their complexity even when a pre-trained version is employed for inferencing purposes. Thus the ‘burden’ of current-generation Transformer-based models is even felt when making inferences. Whereas this inferencing-based concern may not be as potentially debilitating as that relating to training, it is unlikely to be inconsequential. Such concerns may, for example, limit deployment options for inferencing at the edge.

Figure 1: The exponentially escalating complexity of Transformer-based NLP models.
The disruptive impact of Transformer-based models is not restricted to NLP. Application of this attention-centric paradigm for architecting Deep Learning models is already being applied to use cases unrelated to natural language – e.g., in support of computer vision. Transformer-based models underscore the impact of model complexity and extremely large volumes of training data on IT infrastructure – an impact, of course, that applies to use cases of all kinds. With the pressing need to support increasingly autonomous AI use cases (e.g., self-driving vehicles), whose success depends upon input from a plethora of multi-sensory devices, comparable and escalating demands originate from other areas of Deep Learning.11

From NLP to other use cases for Deep Learning, there exists a clear and present situation of exponentially escalating concern; there is literally no option but to refactor the supporting IT infrastructure. In other words, the need for an AI-first infrastructure for Deep Learning is unquestionable. Less obvious, perhaps, is the requirement for a broad and deep toolchain to ensure that any AI-first infrastructure for Deep Learning can be leveraged in practice. To accommodate Transformers alongside existing and emerging use cases for Deep Learning, a tightly integrated ecosystem comprised of both software and hardware is required.
Microsoft has established a reputation for purpose-built IT infrastructures on Azure. Based upon the extremely demanding requirements from NLP to other use cases for Deep Learning, nothing short of a purpose-built IT infrastructure will suffice. For this reason, Microsoft introduced the ND A100 v4 series virtual machines (VMs).

Fundamentally, each ND A100 v4 series VM is based upon eight NVIDIA Ampere A100 Tensor Core GPUs (Figure 2); in isolation, each of these 54-billion transistor GPUs:

- Delivers peak floating point performance metrics in the range of about 10-300 TFLOPS
- Possesses 40 GB of HBM2 GPU memory with 1.6 TB/s of GPU memory bandwidth
- Offers enhanced support for mixed-precision arithmetic plus sparse tensors

Within each VM, pairs of NVIDIA Ampere A100 Tensor Core GPUs are interconnected at 600 GB/s via the NVIDIA NVLINK Bridge; the pairs of GPUs are interconnected via NVIDIA NVswitch (based upon the 64GB/s PCIe Gen4 interconnect). Finally, each VM is populated with an AMD EPYC 7V12 48C CPU.
Without question, each ND A100 v4 series VM is a highly performant compute server. However, to train Transformer-class models for Deep Learning, it is a practical necessity to interconnect ND A100 v4 series VMs. By making use of Mellanox InfiniBand HDR as the interconnect fabric, hundreds of these VMs, and therefore thousands of A100 GPUs, can be interconnected in a fat-tree network topology. Ultimately, this results in a high-bandwidth, low-latency fabric that supports data movement and access between GPUs via NVIDIA GPUDirect RDMA. By making use of NVIDIA NCCL to provide the software primitives required to make optimal use of this fabric, the resulting scale-up-and-out compute cluster can be flexibly leveraged in support of implementing any model for Deep Learning.

It has thus become a practical necessity to make use of ‘supercomputers’ to train Transformer-class models for Deep Learning. Microsoft has acknowledged this need by making generally available AI supercomputers on Azure in four regions. In the most-compelling demonstration to date, the InfiniBand interconnected ND A100 v4 series VMs scaled-up-and-out to achieve top-30 rankings on the TOP500 list of global supercomputers in each of the four regions on Azure. This is a notable first for the public cloud – literally, the fastest public cloud supercomputer ... available on demand.

Six of the top-ten entries on the TOP500 list amplify the case for scale up-and-out IT infrastructure. The ‘hardware archetype’ is quite consistent:

- The scale-up aspect of the system is addressed with accelerators – i.e., recent-generation GPUs such as the NVIDIA Volta GV100, NVIDIA Ampere A100, or NVIDIA Tesla V100
- The scale-out aspect of the system is addressed through CPUs interconnected with recent-generation InfiniBand technology from Mellanox
Whereas the TOP500 emphasizes performance, energy efficiency is the primary concern of the Green500. In the case of the Green500, all the top-ten ranked systems are of the scale up-and-out variety. Of these ten systems, nine make use of NVIDIA Ampere A100 GPUs to scale up in tandem with various CPUs interconnected via InfiniBand technology from Mellanox. In other words, scale-up-and-out IT infrastructures are performant and power-efficient.

It is important to remark that it is clearly possible to craft leadership-class supercomputers from standard components – i.e., NVIDIA GPUs, Mellanox InfiniBand interconnects, and CPUs from almost any vendor. In other words, it is not a requirement to design and fabricate custom-architecture systems to successfully place on leaderboards. This has important ramifications on several levels for purpose-built IT infrastructures:

- The price/performance economics of standard, off-the-shelf components are indisputable and firmly established
- Standard components perform in a consistent fashion regardless of where they are deployed – more precisely, benchmarks obtained in supercomputer deployments on-the-ground are comparable to those realizable in-the-cloud
- Developer productivity is maximized – even as scientists and engineers make use of different platforms for supercomputing, their existing expertise in programming interconnected accelerators can be repeatedly leveraged

And it is to developer productivity that attention shifts momentarily.
AI-First Infrastructure at Any Scale

InfiniBand interconnected ND A100 v4 series VMs are currently the most-compelling demonstration of GPU technology on Azure. Since 2016, Microsoft chose to fashion GPUs into purpose-built configurations in support of visualization (NV-Series GPUs) and computation (NC-Series GPUs). Making computation in support of Deep Learning much more of a point of emphasis, Microsoft introduced ND-Series GPUs on Azure in 2019 and have been steadily expanding GPU-accelerated SKUs ever since. From the obvious to that which is less evident:

- Microsoft has demonstrated a broad-and-deep commitment to GPUs as accelerators of both graphics and compute requirements – this translates to a broad-and-deep matrix of available choices on Azure, and ensures that Deep Learning applications can always target the most-appropriate GPUs (see the “Best options … ” graphic below)

- NVIDIA has evolved GPU technology considerably – though only capturing the past five years, available choices mark progress in not only compute performance and energy efficiency but GPU capabilities (e.g., per-GPU-level interconnection in support of RDMA, varying and dynamically adjustable degrees of precision, support for tensors)

- The strategic partnership between Microsoft and NVIDIA ensures that the latest advances in GPU technology will experience rapid deployment on Azure – thus ensuring customers of a future-proofed investment in cloud-based GPU technology

- There exists sustained value in repurposing GPUs – thus acknowledging that technology introduced in the past for training purposes (or even visualization), may be better re-purposed for inferencing purposes in the present

- Developer productivity is ‘grandfathered’ over the long term – ensuring that any investment made in learning how to program
GPUs remains relevant knowledge and skills owing to the longevity and widespread support for CUDA and OpenACC

- Interconnecting GPUs has evolved to a priority requirement – from use of specialized interconnects within scale-up configurations, to the more-recent advancement of Mellanox InfiniBand use in scale-up-and-out configurations, communication plays an increasingly significant role in facilitating computation

- Support has been progressively enabled for increasingly demanding use cases in Deep Learning – from inferencing to moderate training, and more recently distributed training by exploiting parallelism inherent in data or the models themselves

Evidently, Azure makes for a compelling upgrade alternative to an on-premises deployment. The flexibility offered by Azure, however, is simply not available on-the-ground. Not only can those with Deep Learning applications choose the GPU technology best suited to the specifics of their use case, but they can also do so without delay – as accelerated computing environments on Azure can be deployed for immediate use in minutes. In addition to short-circuiting lengthy procurement processes for physical resources to be deployed on premises, use on Azure transforms the economics of lifetime CAPEX into a pay-per-use OPEX value proposition – again, a refactored value proposition that does not restrict customers to a specific GPU technology. Thus, as research requirements inevitably change, so can the IT infrastructure that supports it.

In principle then, the IT infrastructure for any scale of Deep Learning applications exists on Azure. As important as this is, it comprises tables stakes from the perspective of those charged with developing, running, and maintaining Deep Learning applications. Thus, in the subsequent section attention shifts to enabling the developers and users of Deep Learning applications on Azure.
Best options for optimal GPU utilization

Real-Time Inferencing
- Intelligent Edge
- NVIDIA Triton Inference Server
- Lightweight GPU devices
  - NV & NVv3 Tesla M60
  - NCs_T4_V3 NVIDIA T4 Tensor Core GPU
  - NC Tesla K80

Batch Inferencing
- NV
- NVIDIA V100 Tensor Core GPU

Basic Training
- NCsv2 Tesla K80

Midrange Training
- ND Tesla P40

Data-Parallel Training
- NDv2 NVIDIA V100 Tensor Core GPU (40GB)

Model-Parallel Training
- ND A100 v4 NVIDIA A100 Tensor Core GPU (40GB)
- NDm A100 v4 NVIDIA A100 Tensor Core GPU (80GB)
Building and Operationalizing Models on AI-First Infrastructure

The Azure Machine Learning (Azure ML) service allows emphasis to be shifted onto developers of Deep Learning applications. After illustrating how Azure ML enables developers regardless of their current level of knowledge and skills, attention shifts to the operationalization of models.

Building Models

The breadth and depth of scale-up plus scale-up-and-out IT infrastructure on Azure is impressive. This being the case, the primary opportunity facing developers becomes that of harnessing the most-appropriate GPU technology – whether that GPU technology is isolated to the shared memory of a single VM or requires GPU-based VMs interconnected by a networking fabric intended to facilitate a distributed computation. Of course, for a given use case, it is understood that this choice is likely to evolve – e.g., as applications of Deep Learning make the transition from prototypes to production deployments. In other words, it is not just the use case itself but its operationalization that factors into motivating a need for flexibility in choice of GPU technology – a need that can be easily addressed on Azure but is almost impossible to counter with in on-premises deployments.

From the developer’s perspective, Azure Machine Learning (Azure ML) is the savviest way to leverage the breadth and depth of GPU technology available on Azure. Moreover, as Azure ML is a Platform-as-a-Service (PaaS) on Azure, the development and operationalization of models are both in scope.

Jupyter Notebooks offer a compelling means for developing Machine and Deep Learning models in Python and other programming languages. Other than the Azure ML ‘branding,’ those familiar with scikit-learn will immediately recognize its use in Figure 3. Because Azure ML provides built-in support for scikit-learn, transitioning model development to the cloud can be as simple as uploading an existing Jupyter Notebook. For those needing to code models for Machine and Deep Learning, Azure ML includes built-in support for:
• Development tools – such as Command-Line Interfaces (CLIs), PyCharm and Visual Studio Code in addition to Jupyter Notebooks
• Programming languages – such as R in addition to Python
• Frameworks – such as PyTorch, TensorFlow, and various others, in addition to scikit-learn

Notably, Azure ML includes support for the Open Neural Network Exchange (ONNX); thus, the interoperability challenge resulting from framework proliferation is addressed on Azure as it is on the ground. In some ways more compelling in the case of a cloud-based deployment, availability of ONNX on Azure allows a model to be developed and pre-trained initially through use of one framework, and then subsequently:

• Deployed for inferencing purposes using a different toolchain
• Refactored and/or further enhanced by making use of a completely different framework

Between them, development tools and frameworks, deliver a powerful means for creating and/or refactoring Machine and Deep Learning models. Recall, however, that a critical aspect of the value they deliver is that of abstraction: programming the framework replaces programming distributed accelerators directly. Programming in the context of PyTorch, scikit-learn, TensorFlow, or similar frameworks, all but removes the need to program the Message Passing Interface (MPI) and CUDA (or OpenACC) in C/C++. In other words, the abstraction afforded developers by frameworks is one of simplified syntax and semantics. Of course, if needed, the lower-level programming of distributed accelerators remains a possibility for making contributions (e.g., GPU-accelerated libraries) in support of applications.
With Azure ML, this notion of abstraction is taken to a much higher level. As Figure 4 illustrates, there exists the potential for a ‘no-code option’ in the development of Machine and Deep Learning applications when use is made of Azure Machine Learning Designer. By making use of a visual authoring canvas, pre-existing modules can be composed to address the requirements of quite sophisticated use cases. In the example illustrated in the figure, the objective is to classify images by making use of the DenseNet model via the PyTorch framework. In principle, this highly abstracted approach for model development removes the coding requirement. Thus, someone without coding knowledge or skills, who appreciates Machine or Deep Learning at more of a conceptual level, can rapidly build models. Such an authoring canvas may also be of value to architects or even experienced developers in cases where they need to rapidly prototype an algorithmic approach or model for consideration. Because Azure Machine Learning Designer allows code written in Python or R to be incorporated into the mix, the offering can also be considered to be of the ‘low-code variety.’
Whereas Azure Machine Learning Designer abstracts the creation of Machine and Deep Learning models from writing code to composing modules, Automated Machine Learning (Automated ML) provides the means for surfacing an optimized model as follows (see Figure 5):

1. Based upon the data provided to it, Automated ML engages in **feature engineering** (e.g., selection, and even generation) to accelerate the development of multiple models
2. From the suite of candidate models employed, Automated ML selects the best-performing model
3. For the selected model, a suite of hyperparameter tuning workload is generated, and Automated ML selects the tuned model offering the highest level of accuracy
Thus, Automated ML applies ‘meta-learning’ to derive an optimized model, from a suite of models, with minimal effort from the developer. Based upon technological breakthroughs from Microsoft Research, Automated ML is already proving effective in classic use cases such as classification, regression, and time-series forecasting. As was the case with Azure Machine Learning Designer, Automated ML need not be perceived as a replacement for hand-coding models; rather, Automated ML can:

- Empower those without programmatic knowledge and skills to develop optimized and sophisticated models
- Objectively inform architects or even experienced developers in cases where they might benefit from different algorithmic approaches in model development

Figure 5: Automated ML in action.

Thus far, attention has been focused on the development of models for Machine and/or Deep Learning. Of course, model development only captures an aspect of a much broader life cycle. To broaden the discussion beyond the phase of creating models, in the following section the operationalization of models via Azure Machine Learning receives attention.
Operationalizing Models

Jupyter Notebooks and IDEs allow developers of Deep Learning applications to run their code. When crafting prototypes of algorithmic implementations to entire models, this simplest of approaches can be effective for training, validation, and inferencing purposes. However, the need to properly operationalize the lifecycle of Deep Learning models can escalate rapidly – even before production implementations are established.

![Azure Machine Learning service diagram](image)

**Figure 6: A simple overview of the Azure Machine Learning service.**

Working progressively from left-to-right in the above figure, the essentials of the service are:

**Data** - Their innate ability to be driven by extremely large volumes of data is what differentiates Deep Learning models. As the figure illustrates, such models may need to leverage data that resides in relational databases to file stores, or remotely stored data that can only be retrieved via API calls. Thus, a critical enabler of the Azure ML service is that of the underlying data platform available on Azure. The introduction of a data lake (e.g., via Azure Data Lake\(^{23}\)), for example, allows data from multiple sources to be aggregated into a single storage namespace. Even though the data can retain its original format (e.g., from unstructured to semi-structured to structured), by localizing it in this fashion via a data lake, models can be more easily trained and validated. The
ability to implement scalable solutions for production use is another benefit of a modern data platform that accounts for all the attributes of Big Data.

**Build and Train Models** – From support for traditional Jupyter Notebooks and IDEs, to the no/low-code option of Azure Machine Learning Designer, and finally to Azure Automated Machine Learning meta-learning model development, Azure Machine Learning provides options for developers. Not only does this allow developers to make use of the option best suited to their skill level (from no-code novice to seasoned professional), it also permits alignment with the developer’s intent.24 Azure Machine Learning clearly broadens and deepens the possibilities for building and training models that serve initially as prototypes and ultimately in production. Additionally, Azure ML allows developers to again leverage ONNX – this time for runtime optimizations as opposed to interoperability between frameworks for Machine and Deep Learning.

**Train Models** – Model training is the most compute intensive aspect of the entire Machine Learning lifecycle. As models increase in complexity, so do the demands placed upon the compute infrastructure required to train them.25 As Azure ML is fully integrated with the GPU computing capabilities available on Azure, model training can be scaled up to take advantage of one or more of these accelerators available from an isolated virtual machine. As complexity and/or data volumes increase, Azure ML allows model training to be scaled up and out across a distributed network of VMs.

**Validate** – Once they have been trained, but prior to their use in production, models require validation. Because containerization is a best practice for deploying and scaling models that will be used in production (see below), it makes sense to introduce this means for packaging the model and all its runtime requirements at this stage. Once available, the containerized implementation of the model can be validated on the data that was retained for this purpose. As noted in the schematic, use of Azure Container Instance is ideally suited to the needs of this validation phase. Moreover, uptake of
containerization at this stage aligns well with DevOps principles that promote a proactive stance towards maintaining models (see below).

**Deploy Models** – Open-source Kubernetes is the de facto standard for orchestrating containers such as the validated pre-trained model envisaged here. The Azure Kubernetes Service (AKS) is an implementation of this ‘middleware’ available on Azure. Through its innate ability to replicate containers, AKS ensures that pre-trained can be scaled in response to demand for inferencing purposes. From the cloud to the edge, uptake of AKS ensures that deployments are secure in addition to being scalable.

**Monitor** – Fundamental to operationalizing models is monitoring. From metrics to logs, Azure ML provides monitoring capabilities that can quantify aspects of the development process, through to production-grade deployments. Because monitoring establishes an unambiguous means for tracking their performance in production, the underpinnings of a proactive approach for maintaining models is established.

**CI/CD** – When it comes to Machine Learning applications, change is inevitable. For example, the need to incorporate more-diverse sets of data may trigger the need for retraining a Deep Learning application. As requirements evolve, so will the code, algorithms, methods, etc., needed to realize the application. Making use of code repositories (e.g., via GitHub) is therefore a best practice – a practice whose value escalates rapidly when a development team is involved (as opposed to an isolated individual). Use of GitHub is a prerequisite for implementing DevOps principles. For example, when changes to code are made, a cycle of Continuous Integration and Continuous Deployment (CI/CD) can be enabled through use of Azure DevOps. As illustrated in the figure, this Azure DevOps integration can trigger the retraining of a model. And assuming the retrained model can be properly validated, subsequent deployment and monitoring follows to complete this valuable DevOps feedback loop.
Taken collectively then, Azure ML operationalizes the development, use, and ongoing maintenance of Machine and Deep Learning applications – in other words, it offers lifecycle management from end-to-end. Although it has not been drawn out here to any degree, Azure ML can be applied to multiple applications. For example, using its innate capability to compose applications into interdependent workflows, software pipelines are a core competence of Azure ML.

From model lifecycle management to pipelined applications, Azure ML delivers state-of-the-art capabilities for Machine Learning Operations (MLOps) in the setting of a public cloud. Even though it possesses enterprise-grade capabilities that may only become relevant as production applications are scaled in response to demand, it is important to note that there is absolutely no downside to adopting Azure ML at outset. In other words, from the earliest stages of model prototyping, it is a best practice to adopt Azure ML. As introduced above, Azure ML offers possibilities for better informing and rapidly advancing models during the earliest stages of development. Moreover, as needs inevitably escalate from modest to increasingly demanding, Azure ML can offer considerable value – end-to-end value in the lifecycle management of pipelined applications.

Deep Learning models can rapidly become ‘involved’ – even though algorithms implemented at each layer can be readily deconstructed. Consequently, it typically becomes a challenge to provide the rationale for inferenced outcomes that originate within Deep Learning models. Thus, the explainability (a dimension of Responsible AI) of Deep Learning models is an increasingly pressing requirement. Responsible AI is a necessarily diverse and interdisciplinary topic that receives significant coverage via Azure ML.
Summary

As an enabling technology, AI is making good on many promises. However, to realize its full potential, the models that need to be developed will need to be based upon even more data. Moreover, as expectations of AI mount, so do the requirements for increasingly sophisticated models. Taken together, AI is the victim of its own success, as both data and sophistication translate to more complex models. In the case of Natural Language Processing (NLP), model complexity is already the reality; other use cases seem destined to follow suite. To ensure timely results for model training, and in some cases even inferencing based upon pre-trained models, isolated servers with multiple GPUs are no longer viable. In fact, it is fair to state that a new frontier that demands an AI-first infrastructure is warranted. This infrastructure is necessarily comprised of interconnected GPUs distributed across a network. The resulting scale-up-and-out IT infrastructure is ideally suited to even the most-demanding Transformer-class models that are increasingly common in NLP. On Microsoft Azure, AI supercomputers have been fashioned by interconnecting VMs that house NVIDIA Ampere A100 Tensor Core GPUs with a Mellanox InfiniBand interconnect fabric.

Fortunately, Azure Machine Learning is ideally suited for the building and operationalization of models. With tools suited to developers of every level, Azure ML allows models to be developed and subsequently executed on GPU-accelerated infrastructures of any scale – for purposes of casual prototyping through to enterprise-grade deployments at scale. NVIDIA and Microsoft have been working together to deliver GPU technology on Azure since 2016. Because this partnership results in a broad and deep array of GPU technology that remains current, Azure is the ideal place to build and operationalize AI, Machine and Deep Learning models of any scale.


3 Briefly, LSTMs are a ‘special kind’ of Recurrent Neural Network (RNN) – i.e., a network possessing one or more loops. Importantly, owing to the presence of these loops, in RNNs it is possible for information to persist – hence the allusion to memory. RNNs can suffer from a potentially debilitating design flaw: updates to the network can fail to propagate owing to vanishingly small gradients. LSTMs address this design flaw inherent in RNNs through architectural refactoring. LSTMs were introduced in a seminal paper by Hochreiter & Schmidhuber (Neural Computation, 9(8), 1735-80, 1979, https://doi.org/10.1162/neco.1997.9.8.1735) has been cited over 50,000 times as of this writing.

4 Transformers were first introduced in the “Attention Is All You Need” research paper by Vaswani et al. (https://dl.acm.org/doi/10.5555/3295222.3295349). In four years, it has been cited almost 30,000 times.

5 Quoting again from “Attention is All You Need”: self-attention is “… an attention mechanism relating different positions of a single sequence in order to compute a representation of the sequence.”


7 K. Wiggers, Microsoft and Nvidia team up to train one of the world’s largest language models, Venture Beat, https://venturebeat.com/2021/10/11/microsoft-and-nvidia-team-up-to-train-one-of-the-worlds-largest-language-models/.

8 This exponential growth is well illustrated in a blog post at https://www.microsoft.com/en-us/research/blog/turing-nlg-a-17-billion-parameter-language-model-by-microsoft/.

9 Explain semi-log with allusion to the Richter scale ...

10 Or 5,638 to be more precise.

11 Perception and autonomous-driven AI represent the third and fourth waves of AI, respectively, in futurist Kai-Fu Lee’s taxonomy (AI Superpowers: China, Silicon Valley, And The New World Order, https://www.aisuperpowers.com/).

12 For additional details regarding the Azure ND A100 v4 series VMs see the documentation at https://docs.microsoft.com/en-us/azure/virtual-machines/nda100-v4-series; NVIDIA Ampere A100 Tensor Core GPUs are detailed at https://www.nvidia.com/en-in/data-center/a100/.

13 TFLOPS refer to a trillion floating point operations per second. For 64-bit floating point precision, the A100 peaks at 9.7 TFLOPS, whereas for the 16-bit FP Tensor Core counterpart this peak value is 312 TFLOPS.

14 For additional details regarding GPUDirect Remote Direct Memory Access (RDMA) see https://developer.nvidia.com/gpudirect.

15 For additional details regarding the NVIDIA Collective Communication Library (NCCL) see https://developer.nvidia.com/nccl.

16 AI supercomputers on Azure ranked 26-29 on the June 2021 edition of the Top500 list; details are available online at https://www.top500.org/lists/top500/list/2021/06/.

17 The June 2021 Green500 rankings can be found online at https://www.top500.org/lists/green500/2021/06/.

18 For more information on scikit-learn, see https://scikit-learn.org/stable/index.html.

19 Implicit with built-in support for scikit-learn is the inclusion of NumPy, SciPy, matplotlib, and more. In other words, Azure ML includes the broad and deep development ecosystem that supports Python-based scikit-learn.

20 At this juncture, potential complications relating to data locality have been ignored. As data will be required to train models, this (potentially) non-trivial consideration receives attention elsewhere (appendix/reference???).
21 For the current compilation of built-in support, see https://azure.microsoft.com/en-ca/services/machine-learning/#features.
22 It is important note that a breakthrough from Microsoft Research is responsible for ‘meta-learning’ required to deliver this degree of automation in practice; details on this seminal contribution can be found via arXiv at https://arxiv.org/abs/1705.05355.
24 For example, a highly skilled developer may choose to experiment with a no/low-code option to investigate methods and/or use cases in which they lack expertise. Thus, their ultimate purpose of hand crafting a model is rapidly informed by this exercise.
25 Recall that today, the most-demanding training requirements originate from NLP via transformer-based models.
26 Helm charts allow containers to be managed over their entire life cycle. Helm has been developed to facilitate the creation, installation, and management of applications within Kubernetes. Through use of Helm charts then, the specifics required to update containerized applications deployed in the cloud or at the edge are detailed via a templating mechanism.