A 'Brief' History of Neural Nets and Deep Learning, Part 4

About the mid-2000s neural net comeback, and how deep learning ascended to the tsunami it is today | December 24, 2015

This is the fourth part in 'A Brief History of Neural Nets and Deep Learning'. Parts 1-3 here, here, and here. In this part, we will get to the end of our story and see how deep learning emerged from the slump neural nets found themselves in by the late 90s, and the amazing state of the art results it has achieved since.

> *"Ask anyone in machine learning what kept neural network research alive and they will probably mention one or all of these three names: Geoffrey Hinton, fellow Canadian Yoshua Bengio and Yann LeCun, of Facebook and New York University."*[1]

The Deep Learning Conspiracy

When you want a revolution, start with a conspiracy. With the ascent of Support Vector Machines and the failure of backpropagation, the early 2000s were a dark time for neural net research. LeCun and Hinton variously mention how in this period their papers or the papers of their students were routinely rejected from being published due to their subject being Neural Nets. The above quote is probably an exaggeration - certainly research in Machine Learning and AI was still very active, and other people were also still working with neural nets - but citation counts from the time make it clear that the excitement had leveled off, even if it did not completely evaporate. Still, they persevered. And they found a strong ally outside the research realm: The Canadian government. Funding from the Canadian Institute for Advanced Research (CIFAR), which encourages basic research without direct application, was what motivated Hinton to move to Canada in 1987, and funded his work afterward. But, the funding was ended in the mid 90s just as sentiment towards neural nets was becoming negative again. Rather than relenting and switching his focus, Hinton fought to continue work on neural nets, and managed to secure more funding from CIFAR as told well in this exemplary piece[1]:

> *"But in 2004, Hinton asked to lead a new program on neural computation. The mainstream machine learning community could not have been less interested in neural nets.*

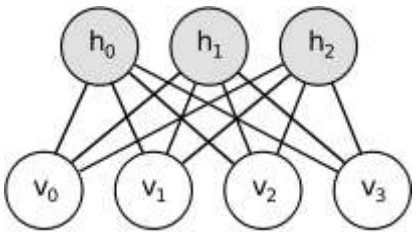
> *"It was the worst possible time," says Bengio, a professor at the Université de Montréal and co-director of the CIFAR program since it was renewed last year. "Everyone else was doing something different. Somehow, Geoff convinced them."*

> *"We should give (CIFAR) a lot of credit for making that gamble."*

> *CIFAR "had a huge impact in forming a community around deep learning," adds LeCun, the CIFAR program's other co-director. "We were outcast a little bit in the broader machine learning community: we couldn't get our papers published. This gave us a place where we could exchange ideas.""*

The funding was modest, but sufficient to enable a small group of researchers to keep working on the topic. As Hinton tells it, they hatched a conspiracy: "rebrand" the frowned-upon field of neural nets with the moniker "Deep Learning" [1]. Then, what every researcher must dream of actually happened: Hinton, Simon Osindero, and Yee-Whye Teh published a paper in 2006 that was seen as a breakthrough, a breakthrough significant enough to rekindle interest in neural nets: **A fast learning algorithm for deep belief nets**[2]. Though, as we'll see, the approaches used in the paper have been superceded by newer work, the movement that is 'Deep Learning' can very persuasively be said to have started precisely with this paper. But, more important than the name was the idea - that neural networks with many layers really could be trained well, if the weights are initialized in a clever way rather than randomly. Hinton once expressed the need for such an advance at the time:

> *"Historically, this was very important in overcoming the belief that these deep neural networks were no good and could never be trained. And that was a very strong belief. A friend of mine sent a paper to ICML [International Conference on Machine Learning], not that long ago, and the referee said it should not accepted by ICML, because it was about neural networks and it was not appropriate for ICML. In fact if you look at ICML last year, there were no papers with 'neural' in the title accepted, so ICML should not accept papers about neural networks. That was only a few years ago. And one of the IEEE journals actually had an official policy of [not accepting your papers]. So, it was a strong belief."*

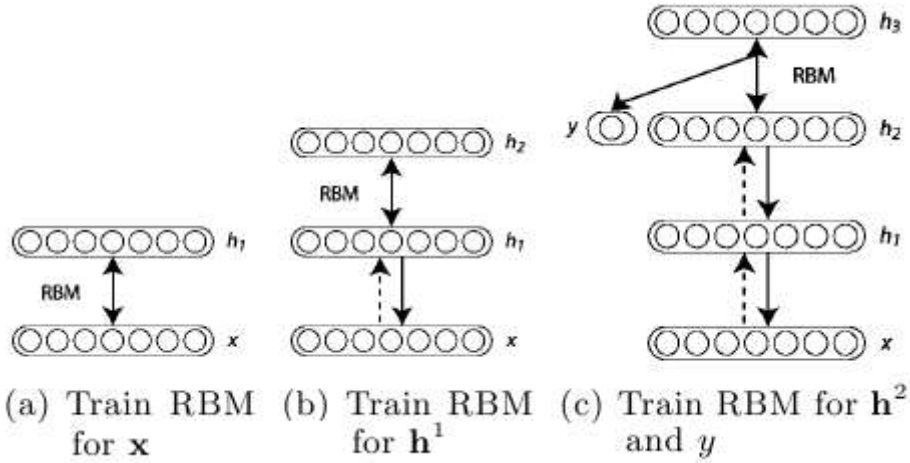


A Restricted Boltzmann Machine. (Source)

So what was the clever way of initializing weights? The basic idea is to train each layer one by one with unsupervised training, which starts off the weights much better than just giving them random values, and then finishing with a round of supervised learning just as is normal for neural nets. Each layer starts out as a Restricted Boltzmann Machine (RBM), which is just a Boltzmann Machine without connections between hidden and visible units as illustrated above, and is taught a generative model of data in an unsupervised fashion. It turns out that this form of Boltzmann machine can be trained in an efficient manner introduced by Hinton in the 2002 "Training Products of Experts by Minimizing Contrastive Divergence"[3]. Basically, this algorithm maximizes something other than the probability of the units generating the training data, which allows for a nice approximation and turns out to still work well. So, using this method, the algorithm is as such:

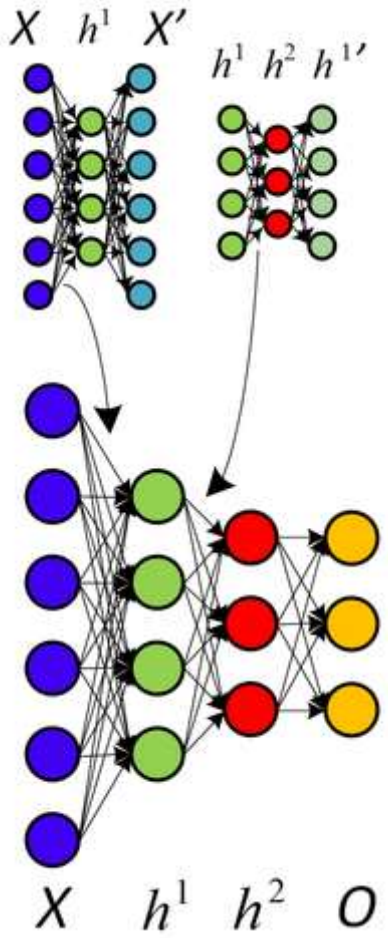
1. Train an RBM on the training data using contrastive-divergence. This is the first layer of the belief net.

2. Generate the hidden values of the trained RBM for the data, and train another RBM using those hidden values. This is the second layer - 'stack' it on top of the first and keep weights in just one direction to form a belief net.
3. Keep doing step 2 for as many layers as are desired for the belief net.
4. If classification is desired, add a small set of hidden units that correspond to the classification labels and do a variation on the wake-sleep algorithm to 'fine-tune' the weights. Such combinations of unsupervised and supervised learning are often called **semi-supervised** learning.



The layerwise pre-training that Hinton introduced. (Source)

The paper concluded by showing that deep belief networks (DBNs) had state of the art performance on the standard MNIST character recognition dataset, significantly outperforming normal neural nets with only a few layers. Yoshua Bengio et al. followed up on this work in 2007 with "Greedy Layer-Wise Training of Deep Networks"[4], in which they present a strong argument that deep machine learning methods (that is, methods with many processing steps, or equivalently with hierarchical feature representations of the data) are more efficient for difficult problems than shallow methods (which two-layer ANNs or support vector machines are examples of).



Another view of unsupervised pre-training, using autoencoders instead of RBMs. (Source)

They also present reasons for why the addition of unsupervised pre-training works, and conclude that this not only initializes the weights in a more optimal way, but perhaps more importantly leads to more useful learned representations of the data. In fact, using RBMs is not that important - unsupervised pre-training of normal neural net layers using backpropagation with plain Autoencoders layers proved to also work well. Likewise, at the same time another approach called Sparse Coding also showed that unsupervised feature learning was a powerful approach for improving supervised learning performance.

So, the key really was having many layers of computing units so that good high-level representation of data could be learned - in complete disagreement with the traditional approach of hand-designing some nice feature extraction steps and only then doing learning using those features. Hinton and Bengio's work had empirically demonstrated that fact, but more importantly, showed the premise that deep neural nets could not be trained well to be false. This, LeCun had already demonstrated with CNNs throughout the 90s, but neural nets still went out of favor. Bengio, in collaboration with Yann LeCun, reiterated this on "Scaling Algorithms Towards AI"[5]:

> *"Until recently, many believed that training deep architectures was too difficult an optimization problem. However, at least two different approaches have worked well in training such architectures: simple gradient descent applied to convolutional networks [LeCun et al., 1989, LeCun et al., 1998] (for signals and images), and more recently, layer-by-layer unsupervised learning followed by gradient descent [Hinton et al., 2006, Bengio et al., 2007, Ranzato et al., 2006]. Research on deep architectures is in its infancy, and better learning algorithms for deep architectures remain to be discovered. Taking a larger perspective on the*

> *objective of discovering learning principles that can lead to AI has been a guiding perspective of this work. We hope to have helped inspire others to seek a solution to the problem of scaling machine learning towards AI."*

And inspire they did. Or at least, they started; though deep learning had not yet gained the tsumani momentum that it has today, the wave had unmistakably begun. Still, the results at that point were not that impressive - most of the demonstrated performance in the papers up to this point was for the MNIST dataset, a classic machine learning task that had been the standard benchmark for algorithms for about a decade. Hinton's 2006 publication demonstrated a very impressive error rate of only 1.25% on the test set, but SVMs had already gotten an error rate of 1.4%, and even simple algorithms could get error rates in the low single digits. And, as was pointed out in the paper, Yann LeCun already demonstrated error rates of 0.95% in 1998 using CNNs.

So, doing well on MNIST was not necessarily that big a deal. Aware of this and confident that it was time for deep learning to take the stage, Hinton and two of his graduate students, Abdel-rahman Mohamed and George Dahl, demonstrated their effectiveness at a far more challenging AI task: Speech Recognition[6]. Using DBNs, the two students and Hinton managed to improve on a decade-old performance record on a standard speech recognition dataset. This was an impressive achievement, but in retrospect seems like only a hint at what was coming - in short, many more broken records.

The Importance of Brute Force

The algorithmic advances described above were undoubtedly important to the emergence of deep learning, but there was another essential component that had emerged in the decade since the 1990s: pure computational power. Following Moore's law, computers got dozens of times faster since the slow days of the 90s, making learning with large datasets and many layers much more tractable. But even this was not enough - CPUs were starting to hit a ceiling in terms of speed growth, and computer power was starting to increase mainly through weakly parallel computations with several CPUs. To learn the millions of weights typical in deep models, the limitations of weak CPU parallelism had to be left behind and replaced with the massively parallel computing powers of GPUs. Realizing this is, in part, how Abdel-rahman Mohamed, George Dahl, and Geoff Hinton accomplished their record breaking speech recognition performance[7]:

> *"Inspired by one of Hinton's lectures on deep neural networks, Mohamed began applying them to speech - but deep neural networks required too much computing power for conventional computers – so Hinton and Mohamed enlisted Dahl. A student in Hinton's lab, Dahl had discovered how to train and simulate neural networks efficiently using the same high-end graphics cards which make vivid computer games feasible on personal computers.*

> *They applied the same method to the problem of recognizing fragments of phonemes in very short windows of speech," said Hinton. "They got significantly better results than previous methods on a standard three-hour benchmark.""*

It's hard to say just how much faster using GPUs over CPUs was in this case, but the paper "Large-scale Deep Unsupervised Learning using Graphics Processors"[8] of the same year suggests a number: 70 times faster. Yes, 70 times - reducing weeks of work into days, even a

single day. The authors, who had previously developed Sparse Coding, included the prolific Machine Learning researcher Andrew Ng, who increasingly realized that making use of lots of training data and of fast computation had been greatly undervalued by researchers in favor of incremental changes in learning algorithms. This idea was strongly supported by 2010's "Deep Big Simple Neural Nets Excel on Handwritten Digit Recognition"[9] (notably co-written by J. Schmidhuber, one of the inventors of the recurrent LTSM networks), which showed a whopping %0.35 error rate could be achieved on the MNIST dataset without anything more special than really big neural nets, a lot of variations on the input, and efficient GPU implementations of backpropagation. These ideas had existed for decades, so although it could not be said that algorithmic advancements did not matter, this result did strongly support the notion that the brute force approach of big training sets and fast parallelized computations were also crucial.

Dahl and Mohamed's use of a GPU to get record breaking results was an early and relatively modest success, but it was sufficient to incite excitement and for the two to be invited to intern at Microsoft Research[1]. Here, they would have the benefit from another trend in computing that had emerged by then: Big Data. That loosest of terms, which in the context of machine learning is easy to understand - lots of training data. And lots of training data is important, because without it neural nets still did not do great - they tended to overfit (perfectly work on the training data, but not generalize to new test data). This makes sense - the complexity of what large neural nets can compute is such that a lot of data is needed to avoid them learning every little unimportant aspect of the training set - but was a major challenge for researchers in the past. So now, the computing and data gathering powers of large companies proved invaluable. The two students handily proved the power of deep learning during their three month internship, and Microsoft Research has been at the forefront of deep learning speech recognition ever since.

Microsoft was not the only BigCompany to recognize the power of deep learning (though it was handily the first). Navdeep Jaitly, another student of Hinton's, went off to a summer internship at Google in 2011. There, he worked on Google's speech recognition, and showed their existing setup could be much improved by incorporating deep learning. The revised approach soon powered Android's speech recognition, replacing much of Google's carefully crafted prior solution [1].

Besides the impressive effects of humble PhD interns on these gigantic companies' products, what is notable here is that both companies were making use of the same ideas - ideas that were out in the open for anyone to work with. And in fact, the work by Microsoft and Google, as well as IBM and Hinton's lab, resulted in the impressively titled "Deep Neural Networks for Acoustic Modeling in Speech Recognition: The Shared Views of Four Research Groups"[10] in 2012. Four research groups - three from companies that could certainly benefit from a briefcase full of patents on the emerging wonder technology of deep learning, and the university research group that popularized that technology - working together and publishing their results to the broader research community. If there was ever an ideal scenario for industry adopting an idea from research, this seems like it.

Not to say the companies were doing this for charity. This was the beginning of all of them exploring how to commercialize the technology, and most of all Google. But it was perhaps not Hinton, but Andrew Ng who incited the company to become likely the world's biggest commercial adopter and proponent of the technology. In 2011, Ng incidentally met with the legendary Googler Jeff Dean while visiting the company, and chatted about his efforts to train

neural nets with Google's fantastic computational resources. This intrigued Dean, and together with Ng they formed Google Brain - an effort to build truly giant neural nets and explore what they could do. The work resulted in unsupervised neural net learning of an unprecedented scale - 16,000 CPU cores powering the learning of a whopping 1 billion weights (for comparison, Hinton's breakthrough 2006 DBN had about 1 million weights). The neural net was trained on Youtube videos, entirely without labels, and learned to recognize the most common objects in those videos - leading of course to the internet's collective glee over the net's discovery of cats:

Google's famous neural-net learned cat. This is the optimal input to one of the neurons. (Source)

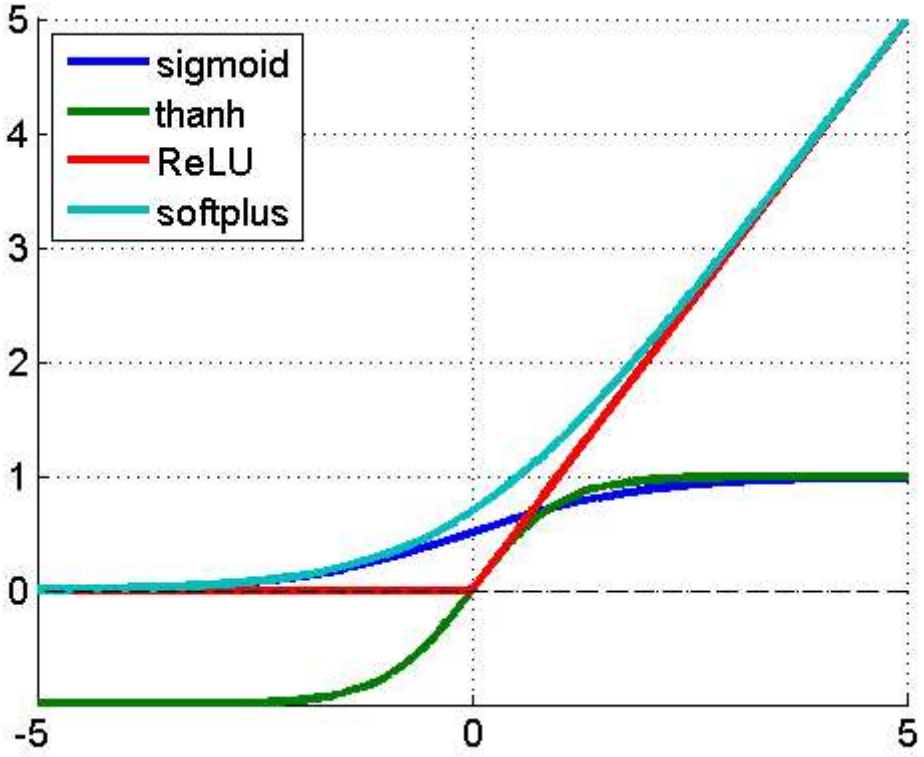
Cute as that was, it was also useful. As they reported in a regularly published paper, the features learned by the model could be used for record setting performance on a standard computer vision benchmark[11]. With that, Google's internal tools for training massive neural nets were born, and they have only continued to evolve since. The wave of deep learning research that began in 2006 had now undeniably made it into industry.

The Ascendance of Deep Learning

While deep learning was making it into industry, the research community was hardly keeping still. The discovery that efficient use of GPUs and computing power in general was so important made people examine long-held assumptions and ask questions that should have perhaps been asked long ago - namely, why exactly does backpropagation not work well? The insight to ask

why the old approaches did not work, rather than why the new approaches did, led Xavier Glort and Yoshua Bengio to write "Understanding the difficulty of training deep feedforward neural networks" in 2010[12]. In it, they discussed two very meaningful findings:

1. The particular non-linear activation function chosen for neurons in a neural net makes a big impact on performance, and the one often used by default is not a good choice.
2. It was not so much choosing random weights that was problematic, as choosing random weights without consideration for which layer the weights are for. The old vanishing gradient problem happens, basically, because backpropagation involves a sequence of multiplications that invariably result in smaller derivatives for earlier layers. That is, unless weights are chosen with difference scales according to the layer they are in - making this simple change results in significant improvements.



Different activation functions. The ReLU is the **rectified linear unit**. (Source)

The second point is quite clear, but the first opens the question: 'what, then, is the best activation function'? Three different groups explored the question (a group with LeCun, with "What is the best multi-stage architecture for object recognition?"[13], a group with Hinton, in "Rectified linear units improve restricted boltzmann machines"[14], and a group with Bengio -"Deep Sparse Rectifier Neural Networks"[15]), and they all found the same surprising answer: the very much non-differentiable and very simple function f(x)=max(0,x) tends to be the best. Surprising, because the function is kind of weird - it is not strictly differentiable, or rather is not differentiable precisely at zero, so on paper as far as math goes it looks pretty ugly. But, clearly the zero case is a pretty small mathematical quibble - a bigger question is why such a simple function, with constant derivatives on either side of 0, is so good. The answer is not precisely known, but a few ideas seem pretty well established:

1. Rectified activation leads to **sparse** representations, meaning not many neurons actually end up needing to output non-zero values for any given input. In the years leading up to

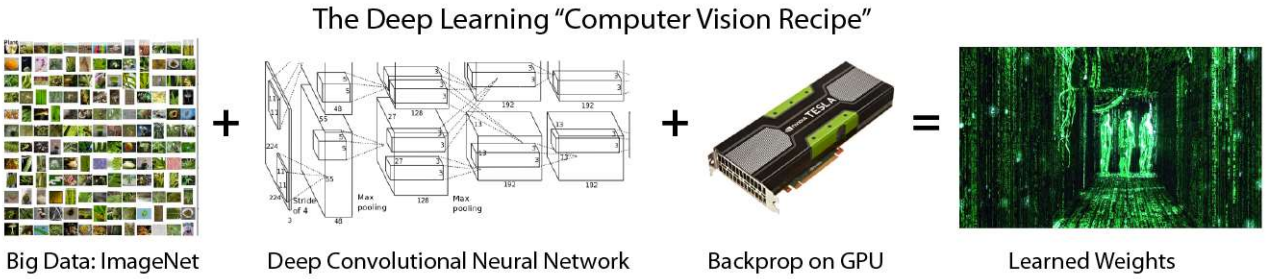
this point sparsity was shown to be beneficial for deep learning, both because it represents information in a more robust manner and because it leads to significant computational efficiency (if most of your neurons are outputting zero, you can in effect ignore most of them and compute things much faster). Incidentally, researchers in computational neuroscience first introduced the importance of sparse computation in the context of the brain's visual system, a decade before it was explored in the context of machine learning.
2. The simplicity of the function, and its derivatives, makes it much faster to work with than the exponential sigmoid or the trigonometric tanh. As with the use of GPUs, this turns out to not just be a small boost but really important for being able to scale neural nets to the point where they perform well on challenging problems.
3. A later analysis titled "Rectifier Nonlinearities Improve Neural Network Acoustic Models" [16], co-written by Andrew Ng, also showed the constant 0 or 1 derivative of the ReLU not too detrimental to learning. In fact, it helps avoid the vanishing gradient problem that was the bane of backpropagation. Furthermore, beside producing more sparse representations, it also produces more distributed representations - meaning is derived from the combination of multiple values of different neurons, rather than being localized to individual neurons.

At this point, with all these discoveries since 2006, it had become clear that unsupervised pre-training is not essential to deep learning. It was helpful, no doubt, but it was also shown that in some cases well-done, purely supervised training (with the correct starting weight scales and activation function) could outperform training that included the unsupervised step. So, why indeed, did purely supervised learning with backpropagation not work well in the past? Geoffrey Hinton summarized the findings up to today in these four points:

1. Our labeled datasets were thousands of times too small.
2. Our computers were millions of times too slow.
3. We initialized the weights in a stupid way.
4. We used the wrong type of non-linearity.

So here we are. Deep learning. The culmination of decades of research, all leading to this:

> ***Deep Learning =***
> ***Lots of training data + Parallel Computation + Scalable, smart algorithms***



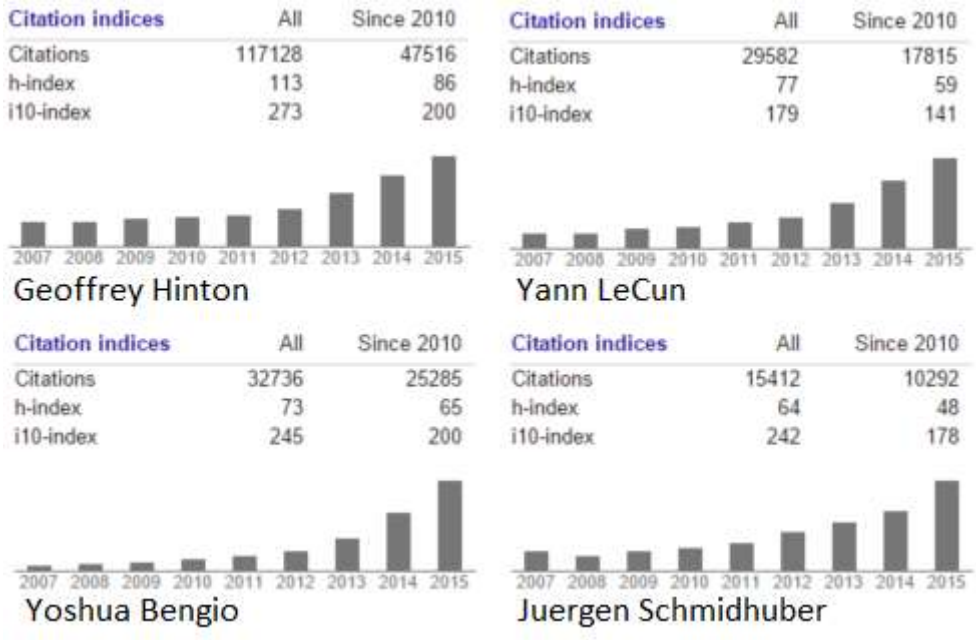
I wish I was first to come up with this delightful equation, but it seems others came up with it before me. (Source)

Not to say all there was to figure out was figured out by this point. Far from it. What had been figured out is exactly the opposite: that peoples' intuition was often wrong, and in particular unquestioned decisions and assumptions were often very unfounded. Asking simple questions,

trying simple things - these had the power to greatly improve state of the art techniques. And precisely that has been happening, with many more ideas and approaches being explored and shared in deep learning since. An example: "Improving neural networks by preventing co-adaptation of feature detectors"[17] by G. E. Hinton et al. The idea is very simple: to prevent overfitting, randomly pretend some neurons are not there while training. This straightforward idea - called **Dropout** - is a very efficient means of implementing the hugely powerful approach of ensemble learning, which just means learning in many different ways from the training data. Random Forests, a dominating technique in machine learning to this day, is chiefly effective due to being a form of ensemble learning. Training many different neural nets is possible but is far too computationally expensive, yet this simple idea in essence achieves the same thing and indeed significantly improves performance.

Still, having all these research discoveries since 2006 is not what made the computer vision or other research communities again respect neural nets. What did do it was something somewhat less noble: completely destroying non-deep learning methods on a modern competitive benchmark. Geoffrey Hinton enlisted two of his Dropout co-writers, Alex Krizhevsky and Ilya Sutskever, to apply the ideas discovered to create an entry to the ILSVRC-2012 computer vision competition. To me, it is very striking to now understand that their work, described in "ImageNet Classification with deep convolutional neural networks"[18], is the combination of very old concepts (a CNN with pooling and convolution layers, variations on the input data) with several new key insight (very efficient GPU implementation, ReLU neurons, dropout), and that this, precisely this, is what modern deep learning is. So, how did they do? Far, far better than the next closest entry: their error rate was %15.3, whereas the second closest was %26.2. This, the first and only CNN entry in that competition, was an undisputed sign that CNNs, and deep learning in general, had to be taken seriously for computer vision. Now, almost all entries to the competition are CNNs - a neural net model Yann LeCun was working with since 1989. And, remember LSTM recurrent neural nets, devised in the 90s by Sepp Hochreiter and Jürgen Schmidhuber to solve the backpropagation problem? Those, too, are now state of the art for sequential tasks such as speech processing.

This was the turning point. A mounting wave of excitement about possible progress has culminated in undeniable achievements that far surpassed what other known techniques could manage. The tsunami metaphor that we started with in part 1, this is where it began, and it has been growing and intensifying to this day. Deep learning is here, and no winter is in sight.



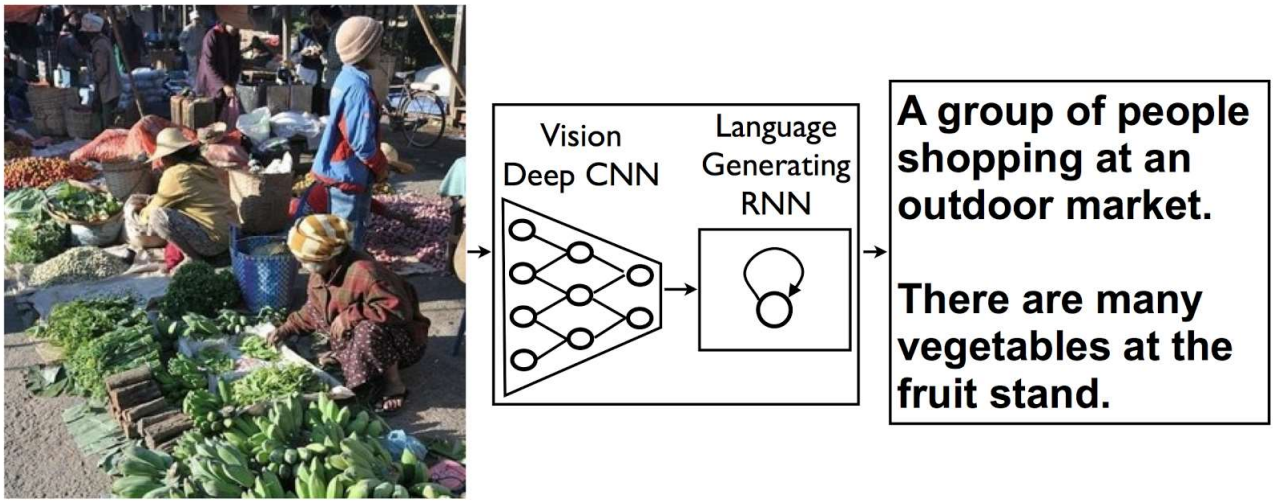
The citation counts for some of the key people we have seen develop deep learning. I believe I don't need to point out the exponential trends since 2012. From Google Scholar.

Epilogue: state of the art

If this were a movie, the 2012 ImageNet competition would likely have been the climax, and now we would have a progression of text describing 'where are they now'. Yann LeCun - Facebook. Geoffrey Hinton - Google. Andrew Ng - Coursera, Google, Baidu. Bengio, Schmidhuber, and Hochreiter actually still in academia - but presumably with many more citations and/or grad students[19]. Though the ideas and achievements of deep learning are definitely exciting, while writing this I was inevitably also moved that these people, who worked in this field for decades (even as most abandoned it), are now rich, successful, and most of all better situated to do research than ever. All these peoples' ideas are still very much out in the open, and in fact basically all these companies are open sourcing their deep learning frameworks, like some sort of utopian vision of industry-led research. What a story.

I was foolish enough to hope I could fit a summary of the most impressive results of the past several years in this part, but at this point it is clear I will not have the space to do so. Perhaps one day there will be a part five of this that can finish out the tale by describing these things, but for now let me provide a brief list:

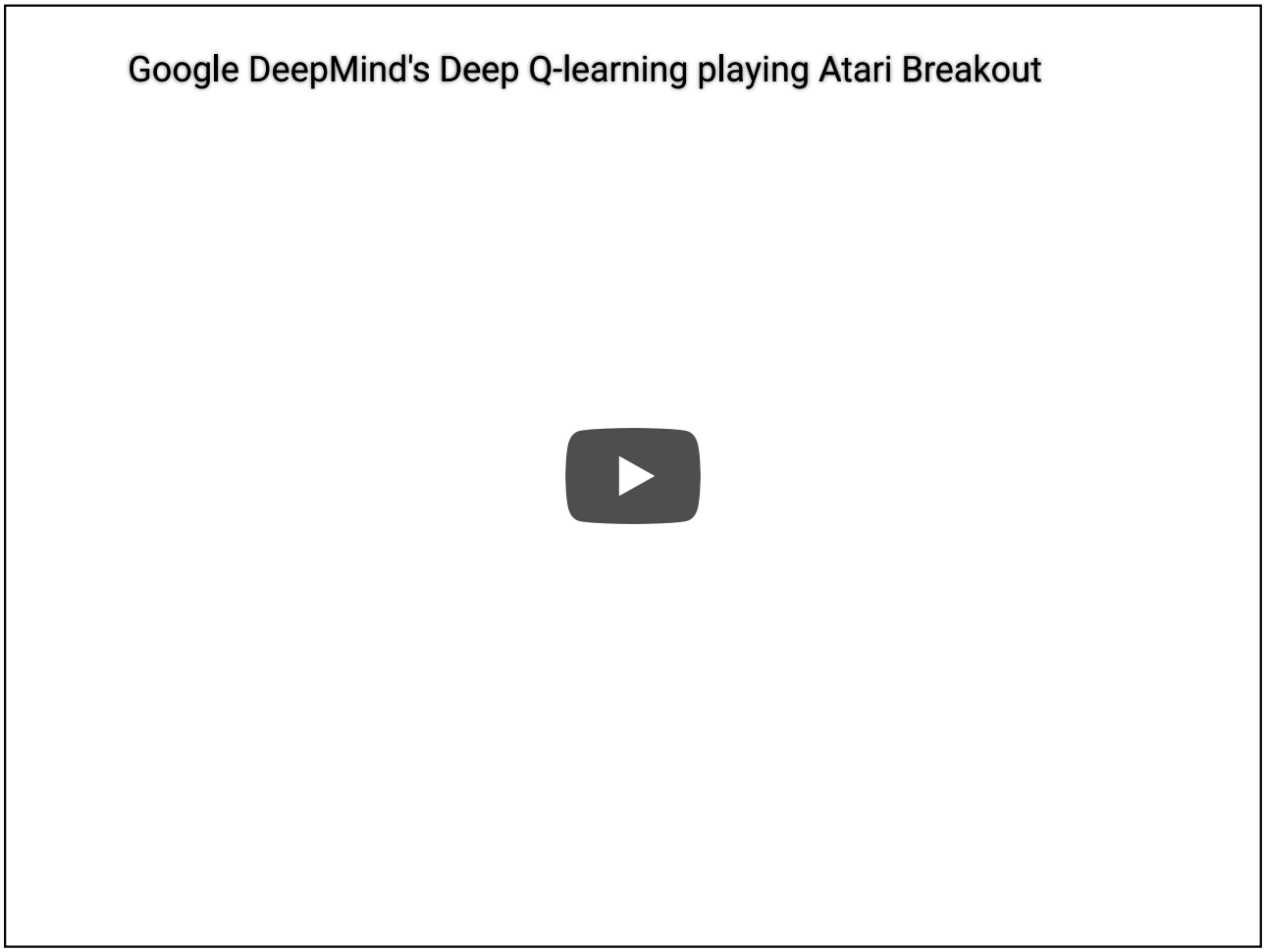
1 - The resurgence of LTSM RNNs + representing 'ideas' with distributed representations



A result from last year. Just look at that! (Source)

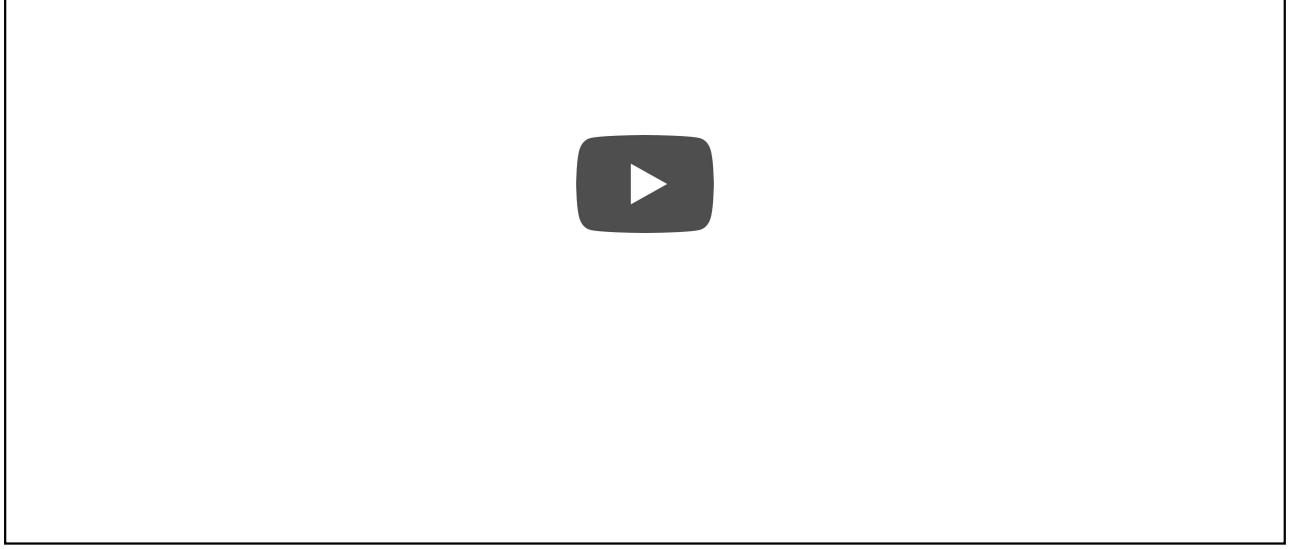
Skype real time translation

2 - Using deep learning for reinforcement learning (again, but better)



Chess playing!

3 - Adding external memory writable and readable to by the neural net



1. Kate Allen. How a Toronto professor's research revolutionized artificial intelligence Science and Technology reporter, Apr 17 2015 http://www.thestar.com/news/world/2015/04/17/how-a-toronto-professors-research-revolutionized-artificial-intelligence.html ↩ ↩2 ↩3 ↩4 ↩5

2. Hinton, G. E., Osindero, S., & Teh, Y. W. (2006). A fast learning algorithm for deep belief nets. Neural computation, 18(7), 1527-1554. ↩

3. Hinton, G. E. (2002). Training products of experts by minimizing contrastive divergence. Neural computation, 14(8), 1771-1800. ↩

4. Bengio, Y., Lamblin, P., Popovici, D., & Larochelle, H. (2007). Greedy layer-wise training of deep networks. Advances in neural information processing systems, 19, 153. ↩

5. Bengio, Y., & LeCun, Y. (2007). Scaling learning algorithms towards AI. Large-scale kernel machines, 34(5). ↩

6. Mohamed, A. R., Sainath, T. N., Dahl, G., Ramabhadran, B., Hinton, G. E., & Picheny, M. (2011, May). Deep belief networks using discriminative features for phone recognition. In Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on (pp. 5060-5063). IEEE. ↩

7. November 26, 2012. Leading breakthroughs in speech recognition software at Microsoft, Google, IBM Source: http://news.utoronto.ca/leading-breakthroughs-speech-recognition-software-microsoft-google-ibm ↩

8. Raina, R., Madhavan, A., & Ng, A. Y. (2009, June). Large-scale deep unsupervised learning using graphics processors. In Proceedings of the 26th annual international conference on machine learning (pp. 873-880). ACM. ↩

9. Claudiu Ciresan, D., Meier, U., Gambardella, L. M., & Schmidhuber, J. (2010). Deep big simple neural nets excel on handwritten digit recognition. arXiv preprint arXiv:1003.0358. ↩

10. Hinton, G., Deng, L., Yu, D., Dahl, G. E., Mohamed, A. R., Jaitly, N., ... & Kingsbury, B. (2012). Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. Signal Processing Magazine, IEEE, 29(6), 82-97. ↩

11. Le, Q. V. (2013, May). Building high-level features using large scale unsupervised learning. In Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on (pp. 8595-8598). IEEE. ↩

12. Glorot, X., & Bengio, Y. (2010). Understanding the difficulty of training deep feedforward neural networks. In International conference on artificial intelligence and statistics (pp. 249-256). ↩

13. Jarrett, K., Kavukcuoglu, K., Ranzato, M. A., & LeCun, Y. (2009, September). What is the best multi-stage architecture for object recognition?. In Computer Vision, 2009 IEEE 12th International Conference on (pp. 2146-2153). IEEE. ↩

14. Nair, V., & Hinton, G. E. (2010). Rectified linear units improve restricted boltzmann machines. In Proceedings of the 27th International Conference on Machine Learning (ICML-10) (pp. 807-814). ↩

15. Glorot, X., Bordes, A., & Bengio, Y. (2011). Deep sparse rectifier neural networks. In International Conference on Artificial Intelligence and Statistics (pp. 315-323). ↩

16. Maas, A. L., Hannun, A. Y., & Ng, A. Y. (2013, June). Rectifier nonlinearities improve neural network acoustic models. In Proc. ICML (Vol. 30). ↩

17. Hinton, G. E., Srivastava, N., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. R. (2012). Improving neural networks by preventing co-adaptation of feature detectors. arXiv preprint arXiv:1207.0580. ↩

18. Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In Advances in neural information processing systems (pp. 1097-1105). ↩

19. http://www.technologyreview.com/news/524026/is-google-cornering-the-market-on-deep-learning/ ↩

Previous Post | Next Post

mark • 11 days ago

A very good series of articles. Thanks.

Francesco Belvedere • 18 days ago

Thank you for this insightful series of articles! Very good overview that really needed to be on the internet for everyone to read.