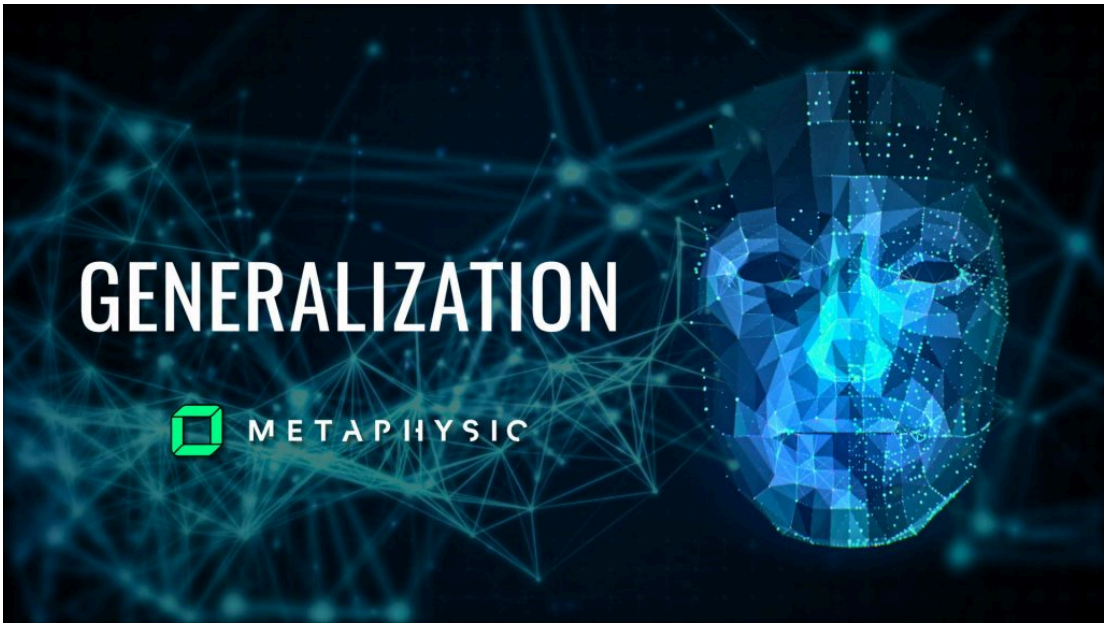
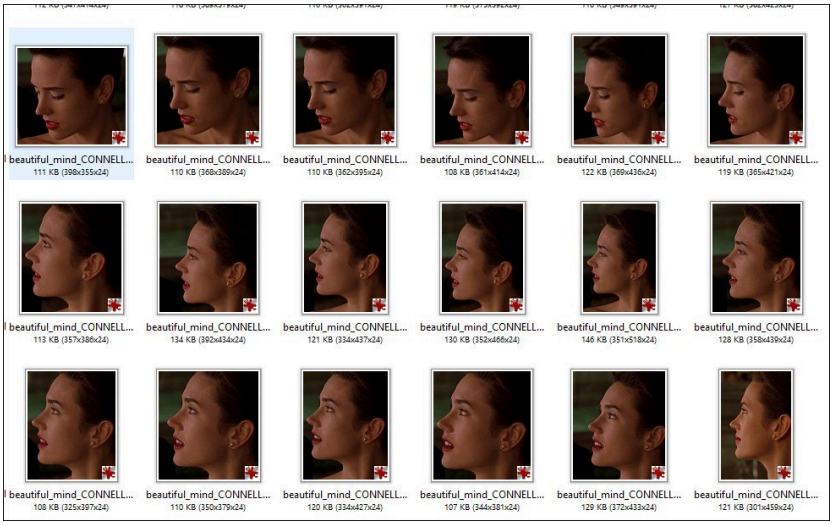
What is Generalization?

Originally published at https://blog.metaphysic.ai/what-is-generalization/

September 15, 2022

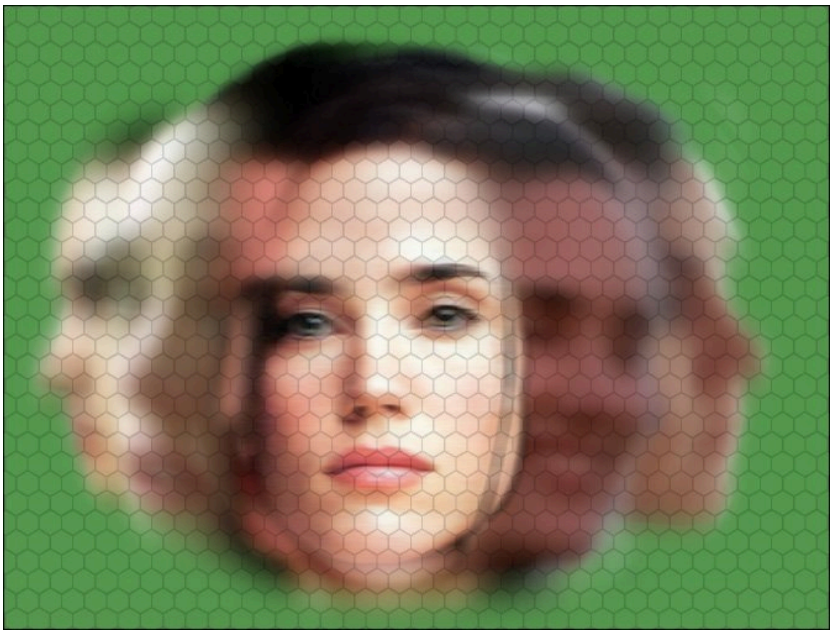


In machine learning, **generalization** is the ability of a trained model to perform effective transformations and inference on data other than the data on which it was trained.
For instance, an autoencoder-based deepfake model is trained on thousands of facial images of a particular person, and its objective is to 'generalize' core concepts from this multiplicity of diverse images into a single neural entity capable of superimposing one particular identity onto the face of another person.



A typical 'face set', in this case for actress Jennifer Connelly, will contain thousands of images in various poses (including diverse facial expressions) and lighting conditions. These images will be passed through the autoencoder's architecture thousands of times, as the system gradually derives a greater understanding of the 'average components' that define this identity in a range of poses and situations.

From each of these thousands of faces, essential numerical features are derived – mathematical vectors which represent the core traits of the training images as numbers instead of pixels.
Since numbers and text are very easy to compress, this means that even a complex and capable deepfake model can weigh less than a gigabyte, since it is storing *no pixel data at all*, but instead the cumulative sum of derived mathematical features for a single person's identity, drawn from thousands of facial images.
As the model is trained, the encoder for the identity that we are training becomes more and more multi-faceted, interconnected and multi-layered, allowing us to extract from it, at any one time and for any one photo or frame, a huge variety of possible poses, facial expressions, angles and lighting conditions – so long as they were included in the training data.
It's not easy to visualize this very abstract, final state of generalization, which is achieved after the model is fully trained. Consider, however, a single person with a thousand faces looking simultaneously in every direction, exhibiting an entire range of emotions in a single frozen moment:



This is a *well-generalized* identity, embedded across the neural network of the trained autoencoder, from which we can draw on a wide range of facets, poses, and dispositions.

Even though the source pictures of Connelly are unlikely to have included the exact lighting in the short target clip for this facial replacement, the style transfer capabilities of autoencoder deepfake packages are capable of matching most ambient lighting environments.

In the image above, the trained model is performing an effective transformation on data that it never saw at training time, but to which it can adapt very well, and can be considered to have achieved functional generalization – at least within reasonable target parameters, as we'll now see.

Obstacles to Effective Generalization

Out-of-distribution (OOD) Data

Generalization is a relative concept. The data, architecture and parameters of any particular machine learning model are usually goal-oriented, and not expected (or designed) to exceed the parameters of its target distribution range.
Thus models of all kinds, from image-based face replacement systems through to GPT-3 style frameworks designed to create complex language responses and extemporizations, will fail – or perform poorly – on out-of-distribution (OOD) data: tasks which exceed the model's design parameters.
For instance, the deepfake demonstration illustrated above works well because the two identities in question share similar facial characteristics, and the model was originally trained on two faces that are also relatively similar to each other (though different enough that the model was forced to develop robust transformative capabilities).
But when the same model is asked to perform a similar transformation on a face with *notably different* facial feature disposition and peripheral bone structure, it adapts as well as it can, but is simply not capable of reconciling the significant mismatch in facial geometry as well as in the earlier case:

For the Jennifer Connelly autoencoder deepfake model (which is the same one used in the above 'Wonder Woman' clip), Scarlett Johannsson's very different facial structure is out-of-distribution (OOD) data as far as the model is concerned.

In most cases, this kind of underperformance represents an *out-of-scope* application rather than a failure of generalization.
Though it's possible in theory to increase the scope of a machine learning model to accommodate a wider range of end-user applications, the data needs, training times and hardware requirements multiply alarmingly as this 'scope creep' rises, and it's usually necessary to pare down the scope and dedicate a full train to the new and altered objective, with or without pre-training or weight importing (see below).

Inadequate Source Data and/or Training

A model can reach optimal generalization and still achieve subpar results, if the amount and/or variety of data is lacking. Training a data-starved or data-imbalanced model to full convergence will not achieve acceptable results in any sector of machine learning research, including facial replacement:

In the case above, a deepfake model was fully trained to convergence, but with large amounts of poor-quality data. The blurred lineaments of the face represent the best sense that the neural network could make of the data.
The same effect can be incurred by stopping training prematurely, before it has reached an optimal convergence, but at which point it may still be useful for lesser tasks, such as face replacements that occupy a small part of a movie frame, where the diminished resolution may be overlooked; where secondary processing or other post-processing procedures can improve the result adequately to meet looming deadlines that protracted training may otherwise have threatened; or where the scope of the system (such as a text synthesis system) is limited enough that greater 'resolution' or versatility is not required for a specific target task.

Overfitting and Fit Training

Overfitting is the opposite of generalization, and occurs when the model, for various reasons, works very effectively on the data that it was trained on, but performs poorly on the 'unseen' data that it was designed to transform and manipulate.
Though overfitting is one of the most frequent curses in computer vision research (where researchers often lack budget to develop datasets at adequate scale to generalize well to larger benchmark datasets), it can be a useful technique in human simulation.
A fairly common version of overfitting, in autoencoder-based deepfake development, is 'fit-training'. In this procedure, one takes a fully-trained and fully-generalized model (i.e. Nick Cage and Bradley Cooper, trained to 2 million iterations for 10 days, and a whopping electricity bill) and continues to train the model with a *completely different* A or B identity (i.e. swapping the Bradley Cooper dataset for a series of exported frames from an Oscar Isaacs movie clip), for a shorter period of time.

The amended model's generalization, particularly for the unchanged identity, suffers a little bit in this process, which is rather similar to ten 'laggard' students joining an advanced math class and blowing its grade curve.
However, the 'switch-hit' new identity can usually form to a high standard very quickly, since it is 'piggy-backing' on the relationships that the model originally developed for the 'abandoned identity. High quality results for the 'interloping' identity can be obtained far, far more quickly than if it had been trained from scratch.
This works best, and is called 'fit training', because the resultant version 2 of the model is not expected to be an 'all rounder', but rather has been given frame data that's specific (i.e. 'fitted') to a target video clip, and can be expected to perform well on that clip – and only on that clip.
You can't rethink the foundations of a building when you're white-walling the fifth floor, and, in all truth, the best results would still be obtained by training from scratch; but the gulp-inducing training times entailed in deep learning model development have made fit-training a popular pursuit in the DeepFaceLab community, and elsewhere.
DFL's chief rival, FaceSwap, similarly offers an option to 'crib' from an expensive, fully-trained model, by importing its weights into a new model, which is effectively the same procedure.

Learning Rates

Depending on how they are set up, models can learn too quickly or too slowly, leading to premature convergence, where the model has raced over the data to a point where it feels it can progress no further – even though the results may be sketchy and nascent, and not at all acceptable.
Else they can 'plateau', often in cases where the learning rate has been set so low that the model becomes 'stuck', and divorced from the wider picture of the gradient that it is supposed to be descending.
Both these factors can impede generalization, and both can benefit from learning rate schedules – a manual or automated rota of adjustments to the learning rate throughout the training schedule, optionally based on the loss value descent as the model training progresses.
One caveat regarding learning rate schedules is that, despite one's best efforts, data tends to vary notably in quality between projects, to the point that a learning rate that achieved successful convergence for an earlier model may actually hinder rather than help your latest build.
As many of the most advanced ML developers concede, there is, sadly, more 'serendipity' and 'luck' needed in this regard than should be the case in such a deterministic pursuit as the training of machine learning models, which is still a surprisingly arcane sphere of study.

Dragged Down by Outlier Data

One final obstacle to successful generalization is the inclusion of enough substandard data to 'drag down' the final converged quality of the model.
Depending on the loss function used during training, the training process itself is always seeking a 'mean average' from which to infer the high-level concepts and classes that will make the model ultimately useful when deployed. To return to our earlier analogy, these are the 'ballast' students that can drag down the whole class, if they are numerous enough.
Including low-quality data that 'fills a gap' where no good-quality data was available (as deepfakers often do with profile training pictures) is a great temptation, but comes at a cost which it's worth considering in advance. The same applies to the addition of new source data at a late stage in training, by which time to core relationships of the network are difficult to reach or to influence, which may lead not only to poor interpretation of the new data, but a reversion to an earlier (higher) overall loss value that may prove difficult to recover again.