Numbers Recognition in Real Time Using Artificial Intelligence

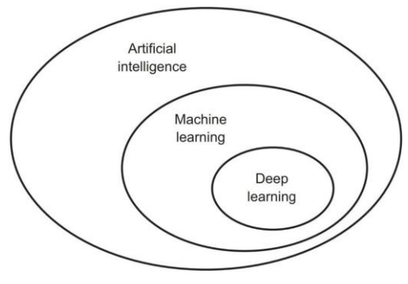
The Team

Girászi Tamás, Rózsa Dominik, Takács Tamás, Giusti Matteo, Mándi Ákos

University of Debrecen: Departments of Informatics

Introduction

Artificial intelligence, also known as AI, and it's subsets such as deep learning are seemingly getting more attention than they used to. Government funding and results have been inconsistent throughout the years, so is it worth to explore and capitalize on the hype?



We are a team of five students from the University of Debrecen pursuing a Bachelor's degree in Computer Science. In our second year of studies we were offered with the possibility of expanding our competences. "Numbers Recognition in Real Time Using Artificial Intelligence" is our firstborn "child" and it is brimming with our achievements and results.

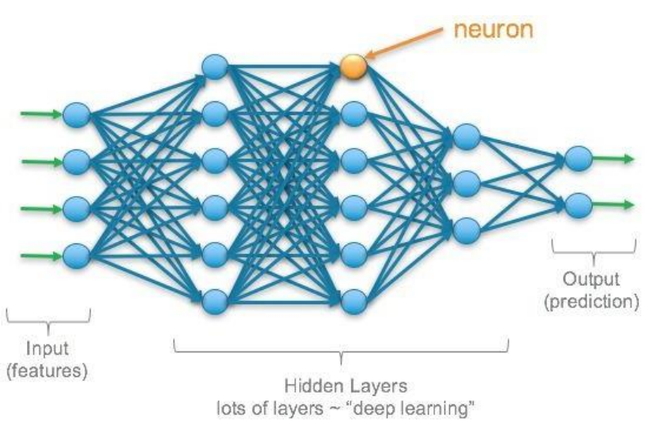
Our group's purpose is to show the world the basics of deep learning and explore the science behind deep neural networks. The end goal was to introduce a program, which can recognize our handwriting and guess the number it's been given in real-time, all while using LabView as front end.

Our Challenges

- Building a convolutional neural network also known as a 'covnet' which can be further applied to our image classification problem.
- Since we are not a big corporation we have limited resources, most importantly less data, thus we have to implement small training datasets.
- We wanted our project to have visual value as well. Manual implementation would be very tedious to look at and would only scare away people who are new to deep learning. Using NI's LabView as a "front end" has been our primary aim ever since the beginning
- Exporting our covnet and implementing it into a VI, has been our biggest challenge yet, since it would make our program visually pleasing and interactive but would come with many difficulties. In the end we would have a program which takes our handwriting as an image, vectorizes it, inserts it into our covnet and through a LabView interface it provides us with the possibilities of which class our written number belongs to. In our case the classes are digits between zero and nine. So how did we do it?

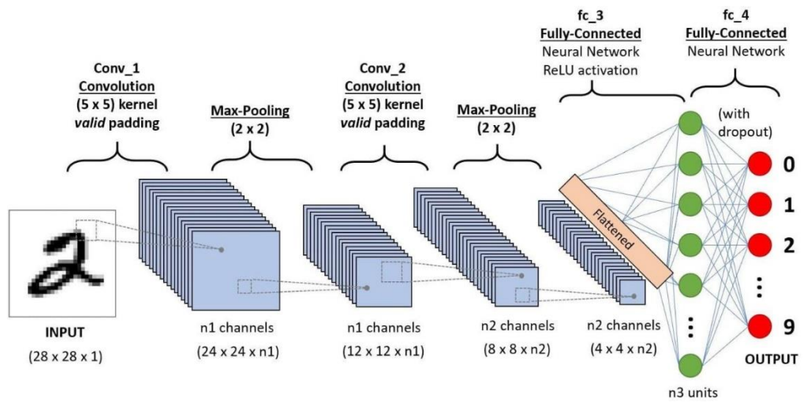
The Method

- Multiclass - classification problems: What are they and how did we approach them
  - Multiclass - classification also known as multinomial classification in deep learning is the problem of classifying instances into one of three or more classes. In our case this extends out to exactly ten classes. In our case this extends out to exactly ten classes, the digits from zero to ten. Deep neural networks use layers, which are the highest - level building blocks of our project. Usually layers are responsible for taking a weighted input, transforming it using non - linear functions and passing these values to the next layer. The proper and optimal set - up of these layers is crucial in producing a suitable output.
- Convolution layer and dense layer: What are the differences and why did we choose covnets?
  - The fundamental difference between a densely connected layer and a convolution layer is that a dense layers learn global patterns in their input feature space, whereas convolution layers learn local patterns such as patterns found in small 2D windows of the inputs. Image these windows as small



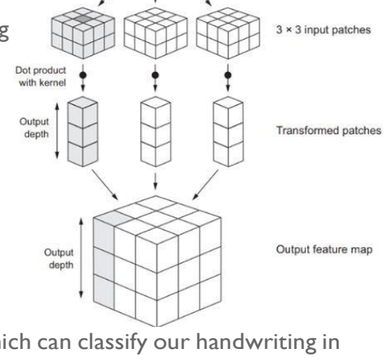
parts of the handwritten image. Using MaxPooling2D and Conv2D we managed to increase our test accuracy by more than 2%, from 97.8% to 99.3% using the MNIST dataset. The upside of convolution layers over dense layers is that the learning is more efficient. Imagine this: Seeing a person in real life for the first time, trying to remember all of its facial features can be quite the challenge, so we use patterns that are easily recognizable and associable to the person of the matter. We use these patterns and recognize them so that when our second encounter happens we would have no problem identifying the person in question. This is exactly what convolution layers do, recognize patterns at exact locations, compared to a dense layer which has to learn everything anew when a new training sample comes along.

- Feature maps: How do we define them and why are they useful?
  - In our project a feature map is a 3D tensor of size (28,28,1) which represents an image's height, width, and depth. Our depth is exactly one since we are using only black and white images. A convolution layer takes a feature map and outputs another feature map of size (28,28,x), where x represents the number of filters over its input. These output channels are basically "response maps" to our actual feature map. This is what the term feature map means; every dimension in the depth axis is a feature and the 2D tensor [:, :, n] is the 2D spatial map of the response of this filter over the input.



- What else did we use?
  - Our base for our projects was Python. We use Jupyter Notebook to build our neural network from scratch to a fully functioning covnet. The dataset we used for training and validation was the MNIST dataset. As students a public dataset was mandatory in the completion of our project. We used Keras with Tensorflow as backend, which are Python libraries used in deep learning, in the making of the neural network.
- What are the remaining building blocks of a deep neural network?
  - Loss functions and optimizers are keys to configuring the learning process. A loss function is the quantity that will be minimized during training. It represents a measure of success for the task at hand. An

optimizer determines how the network will be updated based on the loss function. It implements a specific variant of stochastic gradient descent (SGD)



  - Data preprocessing also takes a huge role in a neural network. It aims at making the raw data at hand more amenable to neural networks. This includes vectorization, normalization, handling missing values and feature extraction.

Overfitting and underfitting also needs to be taken care of. Overfitting happens in every machine learning problem. It proposes a great deal of challenges as to how to interpret optimization and generalization. Our aim is to have our program perform better on data it has never seen before, which is a matter if generalization. Using weight regularization makes simpler models which are less likely to overfit or underfit. Dropout is one of the most effective and commonly used regularization techniques in neural networks.

The Results

We were able to build a covnet suitable enough for our problem with a test accuracy of 99,3% and overall accuracy little below of that. We managed to build a program with the help of LabView which can classify our handwriting in real time. The tools provided were sufficient enough in the making of the user interface, which happened to become user friendly and surprisingly intuitive.

Conclusion

In this project we demonstrated that deep learning truly deserves the hype it's been given by the media. Since the 1950's we have come a long way, through many letdowns and successes, but we as a team will continue to fuel the passion which makes it possible to continue research in the sciences involved in machine learning. In future researches we aim to build a covnets which can recognize whole sentences or mathematical equations in real time.

References

1. Francois Chollet: Deep Learning with Python (2017)
2. Antonio Gulli, Sujit Pal: Deep Learning with Keras (2017)
3. Microsoft Azure Forum
4. Stack Overflow

Acknowledgment

I wish to acknowledge the support of Dr. Bérczes Tamás.

This work was supported by the construction EFOP-3.6.3-VEKOP-16-2017-00002. The project was supported by the European Union, co-financed by the European Social Fund.