Team members

Journal club

Venkata sai Boinepally -700743700 CRN: 23259

V V S Murthy kolla -700729142 CRN:23216

Nirmala Yarlagadda – 700733102 CRN:23216

INJA AKHILESWAR REDDY-700744180 CRN:23216

Video link :

https://drive.google.com/file/d/1OZJtJEHmBePtQOxSn0zSSOFL0zKYkv7V/view?usp=share_link

GitHub link:

https://github.com/niryarjessy22/journal-NNDL.git

Motivation

In our day-to-day life we use many statistical techniques which can be automated, or which can be done by using simple deep learning methods and get more efficient.

Problem statement

In the below papers we will go through different types methodologies

In brief and understand how they are useful in real life scenarios.

Object

paper 1: There are two components in a GAN: (1) a generator and (2) a discriminator. The generator G_θ is a directed latent variable model that deterministically generates samples x from z, and the discriminator D_ϕ is a function whose job is to distinguish samples from the real dataset and the generator.

Paper 2: The Transformer architecture follows an encoder-decoder structure but does not rely on recurrence and convolutions in order to generate an output.

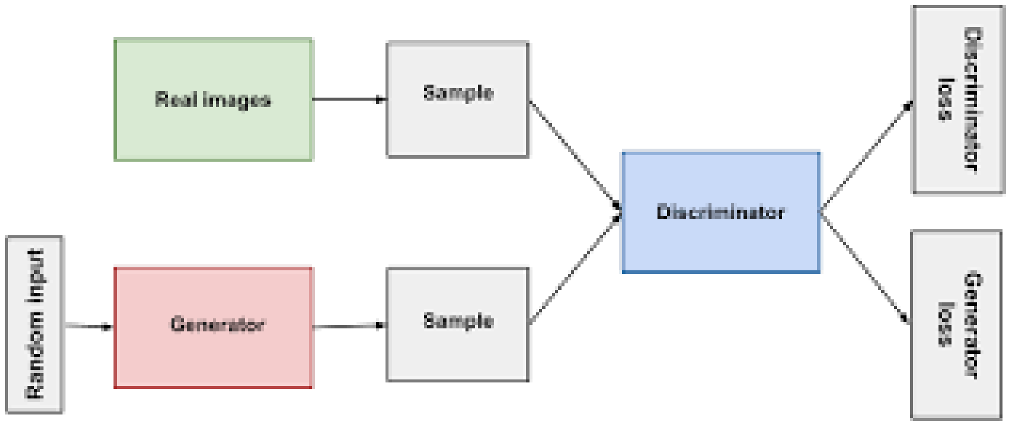
Paper3: The classification of large and complicated images is usually done using Convolutional Neural Network (CNN's)

Paper4: Electroencephalogram (EEG) is a widely used neurophysiology tool. Inspired by the success of deep learning on image representation and neural decoding, we proposed a visual-guided EEG decoding method that contains a decoding stage and a generation stage.

GAN

Generative adversarial networks

- Generative adversarial networks (GANs) are an exciting recent innovation in machine learning. GANs are generative models: they create new data instances that resemble your training data. For example, GANs can create images that look like photographs of human faces, even though the faces don't belong to any real person.
- Generative Adversarial Networks are good at generating random images. As an example, a GAN which was trained on images of cats can generate random images of a cat having two eyes, two ears. But the color pattern on the cat could be very random.
- A generative adversarial network (GAN), which consists of two competing types of deep neural networks, including a generator and a discriminator, has demonstrated remarkable performance in image synthesis and image-to-image translation
- The main focus for GAN (Generative Adversarial Networks) is to generate data from scratch, mostly images but other domains including music have been done. But the scope of application is far bigger than this. Just like the example below, it generates a zebra from a horse.



- A generative model G that captures the data distribution, and a discriminative model D that estimates the probability that a sample came from the training data rather than G. The training procedure for G is to maximize the probability of D making a mistake. This framework corresponds to a minimax two-player game.

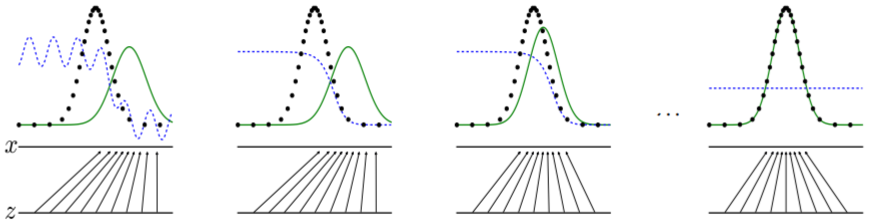
- In the proposed adversarial nets framework, the generative model is pitted against an adversary: a discriminative model that learns to determine whether a sample is from the model distribution or the data distribution. The generative model can be thought of as analogous to a team of counterfeiters, trying to produce fake currency and use it without detection, while the discriminative model is analogous to the police, trying to detect the counterfeit currency

- The adversarial modeling framework is most straightforward to apply when the models are both multilayer perceptrons. To learn the generator's distribution pg over data x, we define a prior on input noise variables pz(z), then represent a mapping to data space as G(z; θg), where G is a differentiable function represented by a multilayer perceptron with parameters θg. We also define a second multilayer perceptron D(x; θd) that outputs a single scalar. D(x) represents the probability that x came from the data rather than pg. We train D to maximize the probability of assigning the correct label to both training examples and samples from G. We simultaneously train G to minimize log(1 − D(G(z))): 2 In other words, D and G play the following two-player minimax game with value function

  V (G, D): min G max D V (D, G) = Ex∼pdata(x) [log D(x)] + Ez∼pz(z) [log(1 − D(G(z)))].

- discriminative is supposed to tell u where there is data and where there is fake data

- In practice, equation 1 may not provide sufficient gradient for G to learn well. Early in learning, when G is poor, D can reject samples with high confidence because they are clearly different from the training data. In this case, log(1 − D(G(z))) saturates. Rather than training G to minimize log(1 − D(G(z))) we can train G to maximize log D(G(z)). This objective function results in the same fixed point of the dynamics of G and D but provides much stronger gradients early in learning.



(a) (b) (c) (d)

- Generative adversarial nets are trained by simultaneously updating the discriminative distribution (D, blue, dashed line) so that it discriminates between samples from the data generating distribution (black, dotted line) px from those of the generative distribution pg (G) (green, solid line). The lower horizontal line is the domain from which z is sampled, in this case uniformly. The horizontal line above is part of the domain of x. The upward arrows show how the mapping x = G(z) imposes the non-uniform distribution pg on transformed samples. G contracts in regions of high density and expands in regions of low density of pg. (a) Consider an adversarial pair near convergence: pg is similar to pdata and D is a partially accurate classifier. (b) In the inner loop of the algorithm D is trained to discriminate samples from data, converging to D ∗ (x) = pdata(x) /pdata(x)+pg(x) . (c) After an update to G, gradient of D has guided G(z) to flow to regions that are more likely to be classified as data. (d) After several steps of training, if G and D have enough capacity, they will reach a point at which both cannot improve because pg = pdata. The discriminator is unable to differentiate between the two distributions, i.e. D(x) = 1 2 .

Disadvantages:

- training these things because you have tarin them in lockstep’s.
- there is no explicit representation.
- You can never build data distribution you can only sample from it.

Advantages :

- mark over chains are never needed
- no inference is needed during learning
- a wide variety of functions can be incorporated into the model

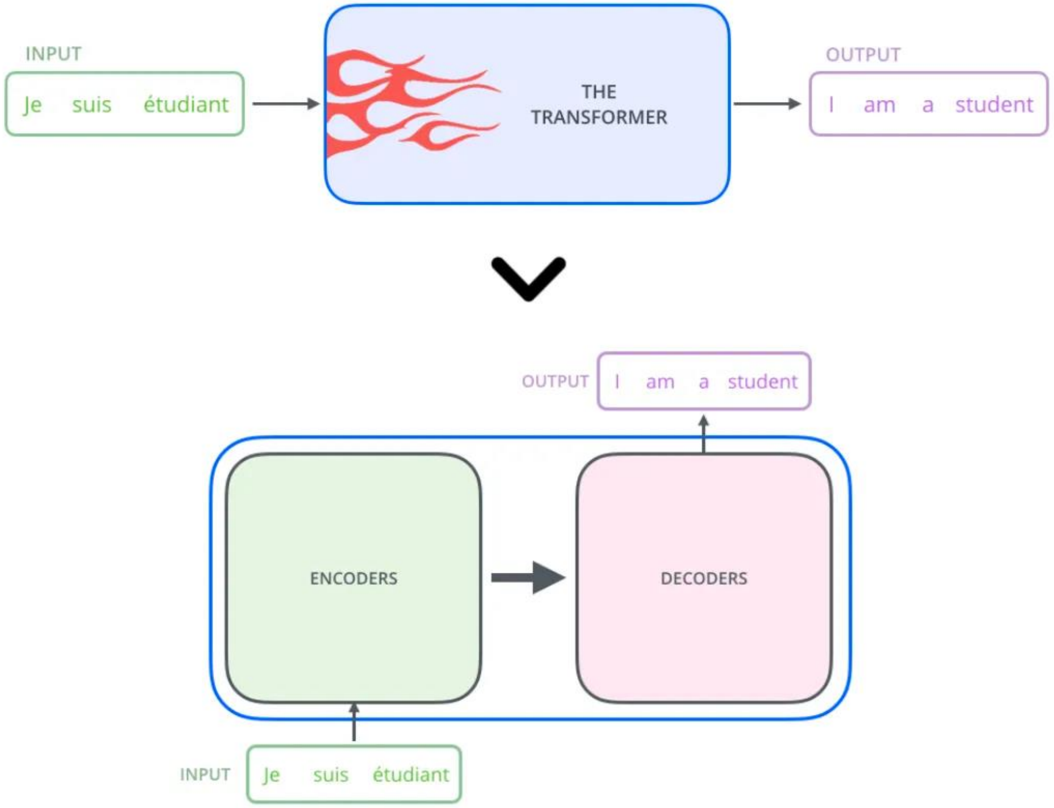
Reference :

https://arxiv.org/pdf/1406.2661.pdf

Transformers

Presentatation by murthy

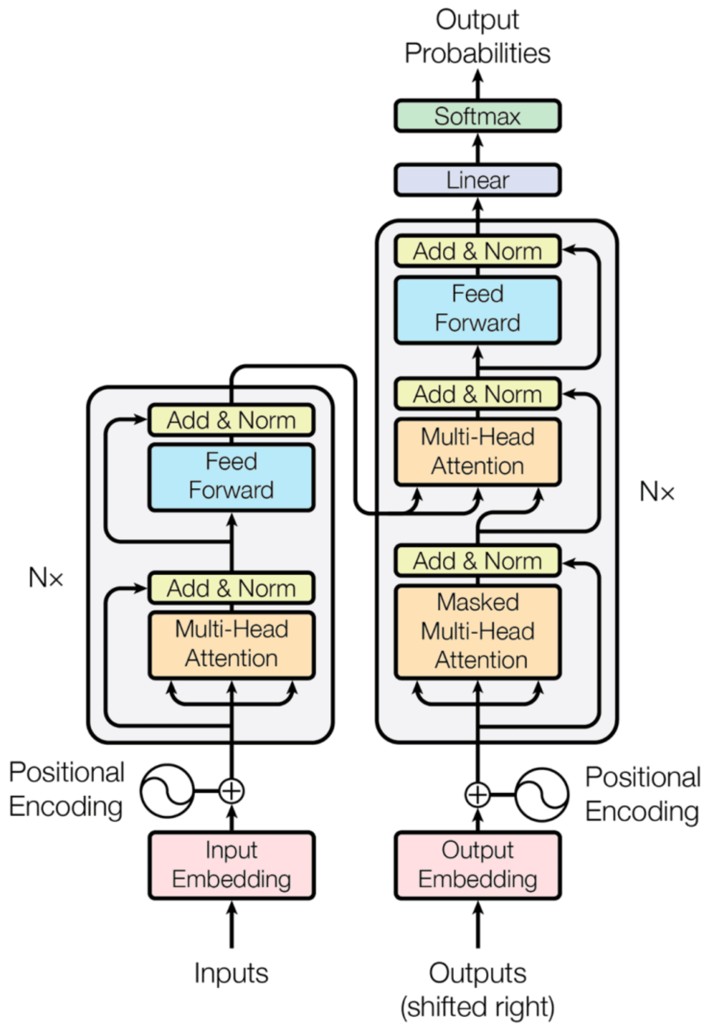
Architecture



A novel architecture called NLP's Transformer aims to solve problems sequentially while resolving long-distance dependencies with ease. It solely relies on self-attention to compute the input and output representations, employing neither convolutions nor sequence-aligned RNNs.

A transformer is a deep learning model that uses the self-attention process and weights the importance of each component of the input data differently.

Attention Mechanism



Basically, when the generalized attention mechanism is provided with a string of words, it evaluates each key in the database using the query vector assigned to a particular word in the string. This depicts the relationship between the word under examination and the other words in the sequence.

Results

Use TensorFlow Datasets to load the [Portuguese-English translation dataset](#)

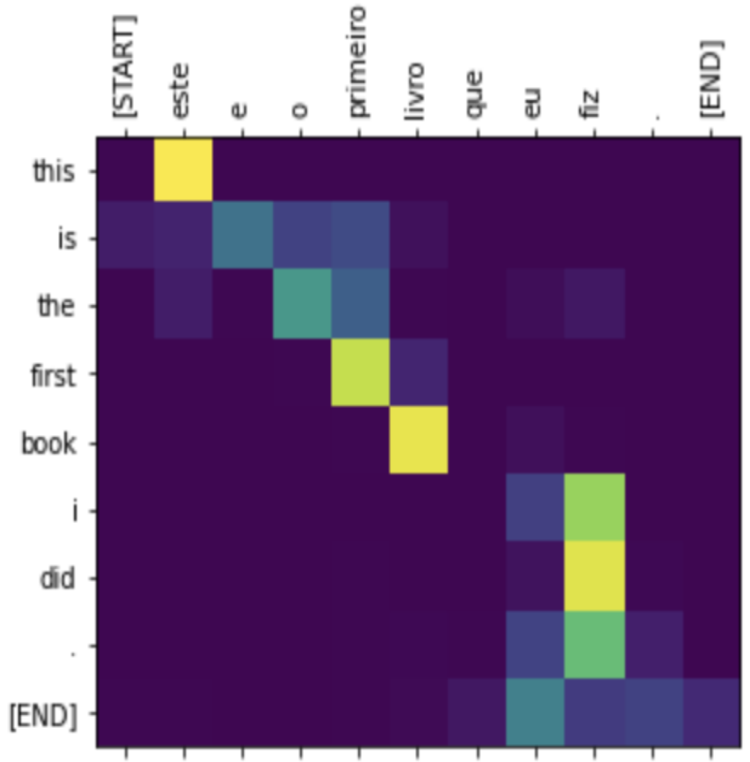
```
transformer.summary()
```

```
Model: "transformer"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 encoder_1 (Encoder)         multiple                  3632768

 decoder_1 (Decoder)         multiple                  5647104

 dense_38 (Dense)            multiple                  904290

=================================================================
Total params: 10,184,162
Trainable params: 10,184,162
Non-trainable params: 0
_________________________________________________________________
```



Reference

ACM journal : https://dl.acm.org/doi/10.5555/3295222.3295349

Image classification using small convolutional neural network

INTRODUCTION

- The technique of classifying photos into several groups according to their content is known as image classification.
- Deep learning architectures known as convolutional neural networks (CNNs) are frequently employed for image categorization problems.
- In this demonstration, we'll utilize a miniature CNN to categorize pictures of cats and dogs.

Small CNN Architecture

- When computer resources are restricted, image classification tasks can be performed using a compact CNN architecture.
- Less convolutional layers, smaller filters, and fewer neurons in the fully linked layers may make up a compact CNN architecture.
- We will use two convolutional layers, two max pooling layers, and two fully connected layers in our tiny CNN to categorize cats and dogs.

Training a Small CNN

- A tiny CNN is trained by feeding it a collection of labeled images and then tweaking the network's weights to reduce the discrepancy between the predicted and actual labels.
- We'll make use of the 25,000 cat and dog photos in the Kaggle Cats and Dogs dataset.
- To train and test our little CNN, the dataset will be divided into training and validation sets.

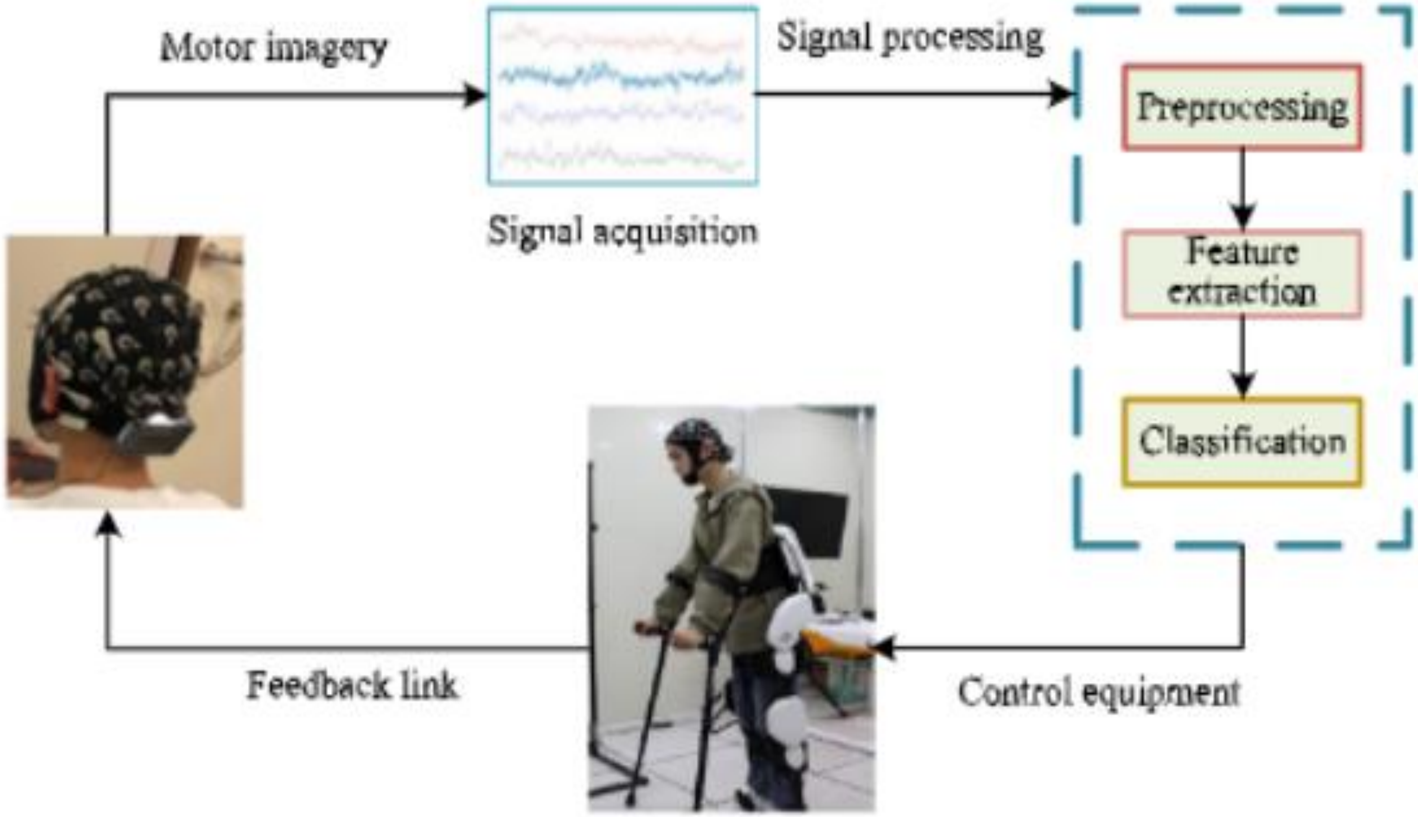
Example Results & Conclusion

- Our tiny CNN achieves an accuracy of 88.6% on the validation set after 20 epochs of training. While some sample photos of cats and dogs and the projected class labels are shown on the right, the training and validation accuracy and loss curves are shown on the left. Most of the images can be correctly classified by our tiny CNN, while there are a few misclassifications.
- We have seen how a simple CNN may be applied to picture classification tasks in this example. Although having a lesser accuracy than larger, more complicated CNNs, our little CNN was nevertheless able to produce useful findings with fewer processing resources. When deploying models on computing-constrained devices like mobile phones or Internet of Things (IoT) gadgets, small CNNs are especially helpful.

EEG decoding method based on multi-feature information fusion for spinal cord injury

- To develop an efficient brain–computer interface (BCI) system, electroencephalography (EEG) measures neuronal activities in different brain regions through electrodes. In this paper, a deep learning framework based on a modified graph convolution neural network (M-GCN) is proposed, in which temporal-frequency processing is performed on the data through modified S-transform (MST) to improve the decoding performance of original EEG signals in different types of MI recognition.

Fig : BCI system structure block diagram



Conclusion and Future work :

- The M-GCN algorithm proposed in this paper has higher classification accuracy than CNN (76.471%),RNN (81.468%) and SVM (59.624%).
- By detecting the ERD/ERS phenomena of each specific rhythm of different MI recognition, the characteristics contained in the EEG signal are also different. The BCI system can effectively identify the EEG signal generated by MI according to the different characteristics, and obtain the movement intention of the subjects

Reference :
https://www.sciencedirect.com/science/article/pii/S0893608022003641/pdfft?md5=f4ea7aeba0e828aaad53df11c96313fd&pid=1-s2.0-S0893608022003641-main.pdf